Tampere University

Joonas Helminen

AUTOMATED GENERATION OF STEEL CONNECTIONS OF BIM BY MACHINE LEARNING

Faculty of Built Environment
Master of Science Thesis
June 2019

ABSTRACT

Joonas Helminen: Automated generation of steel connections of BIM by machine learning
Master of Science Thesis
Tampere University
Master's Degree Programme in Civil Engineering
May 2019

In the last decades, building information modeling (BIM) has increased significantly and it has widely accepted in the construction industry. This has made available a significant amount of digital data that makes possible the use of machine learning techniques in the BIM field. However, machine learning techniques are yet to utilize and current approaches for automation are yet to take full advantage of the information gathered in previously engineered BIM models.

In this study, it was investigated improving modelling efficiency by developing a new toolkit for automated generation of steel connections in BIM models by machine learning techniques. The toolkit had three objectives: generate a training dataset, predict connections between structural members based on the dataset, and automatically model them. The toolkit consists of modules developed in C# and Python, with the machine learning module being implemented using the latter. For this module, the k-nearest neighbors (k-NN) algorithm was used for prediction.

The toolkit was tested on 13 industrial steel structures. Connections were searched and automatically created to three models and a training dataset contained connections from 10 models. The results were positive, even being limited by using only a 10-model database. By creating a training set from finished models, it was found that it is possible to predict and automatically insert valid structural connections in new BIM models. Overall, our findings suggest that our methodology promises to be of significant assistance in improving present methods of generating steel connections in building design.

Keywords: BIM, machine learning, automation, steel connections

TIIVISTELMÄ

Joonas Helminen: Teräsliitosten automaattinen tietomallintaminen koneoppimista käyttäen
Diplomityö
Tampereen yliopisto
Rakennustekniikan diplomi-insinöörin tutkinto-ohjelma
Toukokuu 2019

Tietomallien käyttö rakennusalalla on yleistynyt runsaasti viimeisimpien vuosikymmenten aikana ja niiden käytöstä on tullut osa arkipäiväistä toimintaa. Tämä on mahdollistanut valtavan määrän digitaalisessa muodossa olevaa tietoa, jota voidaan käyttää hyväksi koneoppimismenetelmissä. Kuitenkaan koneoppimismenetelmiä ei ole vielä juurikaan käytetty hyväksi ja nykyiset ratkaisut suunnittelun automatisoimiseksi eivät käytä hyväksi mahdollisuutta koota tietoa aikaisemmin suunnitelluista tietomalleista.

Työssä on selvitetty, kuinka tietomallintamisen tehokkuutta voitaisiin parantaa kehittämällä ohjelma teräsliitosten automaattiseen tietomallintamiseen koneoppimismenetelmiä hyödyntäen. Ohjelmalla oli kolme päämäärää: luoda opetusdatasetti, tyypittää liitokset rakenneosien välillä datasetin perusteella ja mallintaa ne automaattisesti. Ohjelma koostuu C# ja Python- ohjelmointikielillä kehitetyistä moduuleista, joista jälkimmäistä on käytetty koneoppimismoduulin toteuttamiseen. Käytetty koneoppimismenetelmä on k-nearest neighbor (k-NN).

Ohjelma testattiin 13 teräsrankaisella teollisuusrakennusmallilla. Kolmeen tietomalleista etsittiin ja luotiin liitokset automaattisesti ja opetusdatasetti luotiin kymmenestä tietomallista kootuista liitoksista. Tulokset olivat positiivisia, vaikka opetusdatasetti oli rajoitettu ainoastaan 10 tietomalliin. Työn perusteella voidaan sanoa, että liitosten tyypitys ja automaattinen mallintaminen ovat mahdollista suorittaa valmistuneista tietomalleista luodun opetusdatasetin perusteella. Lisäksi, työn pohjalta voidaan ennakoida, että koneoppimismenetelmien käytöllä voidaan merkittävästi parantaa tietomallintamisen tehokkuutta teräsliitosten mallintamisen automatisoinnissa.

Avainsanat: tietomalli, koneoppiminen, automatisointi, teräsliitokset

PREFACE

First of all, I would like to express my gratitude to Sweco Structures Ltd. for making possible to do my master's thesis while working there and for providing opportunity to work with machine learning. I thank all my colleagues who have helped me to complete this project and to those who provided valuable insights for this research. Specifically, I want to thank Ricardo Farinha from Sweco Finland Ltd. for guiding and inspiring.

I would like to show appreciation to my supervisor, Professor Mikko Malaska.

Finally, special thanks to my friends and family, who have encouraged and supported me through the project.

Tampere, 25 June 2019

Joonas Helminen

CONTENTS

1. INTRODUCTION 1
   1.1 Background and motivation 1
   1.2 Research objectives and methods 3
   1.3 Structure of the thesis 3
2. THEORETICAL BACKGROUND 5
   2.1 Introduction to steel connections 5
      2.1.1 Components in steel connections 6
      2.1.2 Connection choosing process 7
      2.1.3 Connection capacity and verification 9
   2.2 Building information modelling 10
      2.2.1 Detailing in BIM 11
      2.2.2 Storing data in BIM and in CAD 13
   2.3 Machine learning 15
      2.3.1 Data types in machine learning 17
      2.3.2 Clustering 18
      2.3.3 Classification 19
      2.3.4 Artificial intelligence in BIM 21
3. IMPLEMENTATION 23
   3.1 Tekla Structures 24
      3.1.1 Automation in Tekla Structures 24
      3.1.2 Tekla Open API 27
   3.2 Machine learning 29
      3.2.1 Creating database 30
      3.2.2 Preprocessing 32
      3.2.3 Algorithm 33
      3.2.4 Reporting results 35
4. RESULTS 37
   4.1 Methods and objectives 37
   4.2 Test cases 37
   4.3 Finding connections 38
   4.4 Automatic modelling 42
   4.5 Quality 42
5. DISCUSSION 44
   5.1 Evaluation 44
   5.2 Limitation and reliability 45
   5.3 Theoretical contribution 45
   5.4 Further research 46
6. CONCLUSIONS 48
REFERENCES 49

LIST OF FIGURES

Figure 1. *Indexes of labor productivity for construction and non-farm industries in the Unit-ed States between 1964 and 2009. (Eastman et al., 2011)* 2

Figure 2. *An example of assembled connection (adapted from Liu et al., 2015)* 5

Figure 3. *Typical beam-to-column connections (Davison & Owens, 2011).* 8

Figure 4. *An example of BIM model in Tekla Structures.* 11

Figure 5. *An example of a connection in a BIM model.* 12

Figure 6. *Simplified interrelation between material, object and assembly (adapted from Weygant, 2011).* 14

Figure 7. *An example of decision tree (Bell, 2014).* 17

Figure 8. *Illustration of clustering. (Left) Two-dimensional input data; (Right) final clustering obtained by K-means algorithm. (Jain, 2010)* 19

Figure 9. *An example of classification process: (a) learning and (b) classification. (Han et al., 2006)* 20

Figure 10. *Architecture of the toolkit.* 23

Figure 11. *Automation in Tekla Structures* 25

Figure 12. *An example of a plugin user interface* 26

Figure 13. *Tekla Open API libraries* 28

Figure 14. *Action in a machine learning project. (Bell, 2014)* 30

Figure 15. *The main structure of the database* 31

Figure 16. *An example of a detected connection area.* 32

Figure 17. *The k-NN binary classification example in two dimensions, where k is 15. (Hastie et al, 2009).* 34

Figure 18. *An example of proposed connections to a connection area.* 36

Figure 19. *Models for finding connections: (a) Model I; (b) Model II; (c) Model III.* 38

Figure 20. *Representation of the results of the toolkit in BIM model I: (Left) Connection areas; (Right) Results presented using colors.* 39

Figure 21. *Histogram and cumulative curve of the percentage of matches for model I: (Left) Perfect matches; (Right) Perfect and possible matches.* 41

Figure 22. *Histogram and cumulative curve of the percentage of matches for model II: (Left) Perfect matches; (Right) Perfect and possible matches.* 41

Figure 23. *Histogram and cumulative curve of the percentage of matches for model III: (Left) Perfect matches; (Right) Perfect and possible matches.* 42

LIST OF SYMBOLS AND ABBREVIATIONS

AEC	Architecture, Engineering and Construction
API	Application Programming Interface
BIM	Building Information Modelling
CAD	Computer Aided Design
DLL	Dynamic Link Library
k-NN	k-Nearest Neighbors
RPC	Remote Procedure Call
UI	User Interface

1. INTRODUCTION

In the last decades, building information modeling (BIM) has increased significantly and it has widely accepted in the construction industry. A survey in Finland showed that 92 % of the participants predicted that they will be using BIM by 2018 (Finne et al., 2013). BIM is a simulation of a building which contains significantly more information on the actual building than drawings produced using the Computer Aided Drafting (CAD) system (Volk et al., 2014) and it has successfully assisted in eliminating faults in designs (Ning & Young, 2010). However, automation is yet to make full usage of the potential efficiency increase (Correa, 2015).

1.1 Background and motivation

The construction industry is renowned for its poor productivity and lags behind other industries in the rate by which improvements are introduced (Fulford and Standing 2014; Segerstedt et al., 2010). This can also be seen in Figure 1 that illustrates how the productivity of construction industry has not been developing while the productivity of other non-farm industries has doubled in the last five decades. The usage of BIM promises to revert this trend but software tools able to implement BIM are not yet fully matured (NIBS, 2007). E.g., according to Oti et al. (2016), plug-ins and external programs are yet to make full usage of the potential applications of BIM.

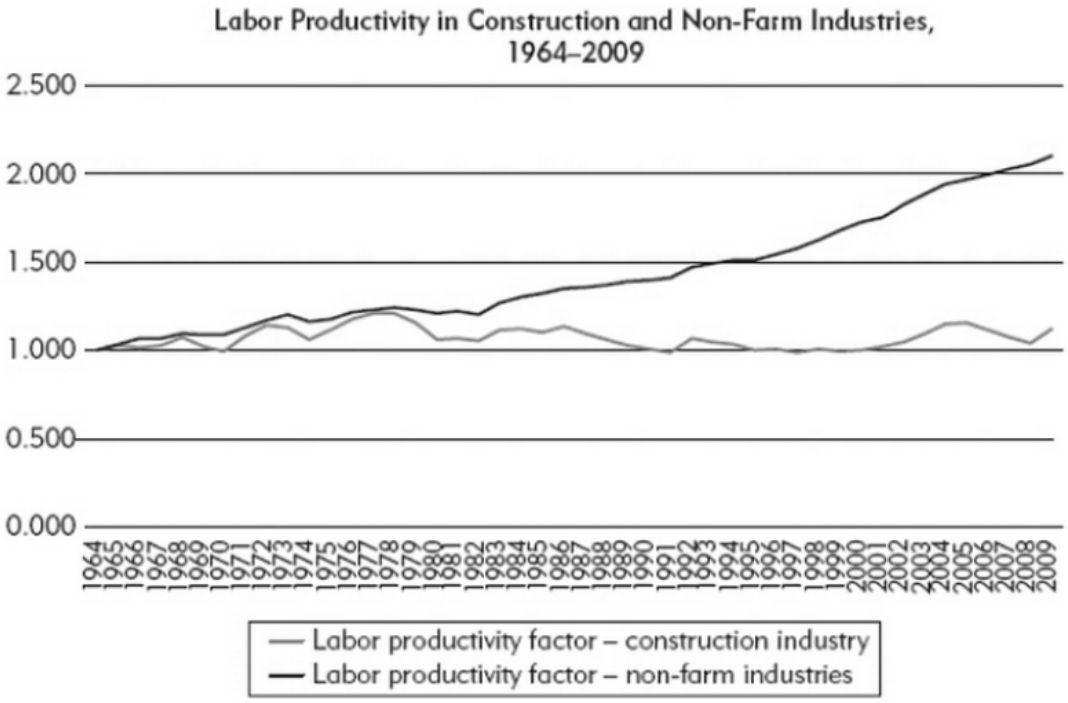

Figure 1. *Indexes of labor productivity for construction and non-farm industries in the Unit-ed States between 1964 and 2009. (Eastman et al., 2011)*

The next innovation to make designing processes more efficient is likely to be its automation (Correa, 2015). In this regard, the automatic generation of BIM models has been explored (Banfi et al., 2017; Eastman et al., 2009; Wang et al., 2015). Present approaches to tackle this problem consist either of generating a BIM model from data on existing buildings or the generation of buildings design from a set of predefined rules. These approaches, while promising, have not yet been able to take full advantage of the information gathered in previously engineered BIM models (Helminen et al., 2018).

Machine learning is widely used in many fields, ranging from Computer Science, to Physics and Biology. However, it's not yet widely researched in BIM or more generally in Architecture, Engineering and Construction (AEC) industry. One reason for this is that the storage of digital information in this area that can be used in quantitative research has only recently been used. Storing this data in digital format is what allows the introduction of data analysis techniques in this area (Jain, 2010). A consequence of this is already visible, in that of the increased amount of BIM is providing more accessible and structured data.

Relevantly, in the AEC industry, the design phase normally accounts for 5-10% of the total cost of a project (Yarmohammadi, 2017). In this regard, Corenc et al. (2015) indicated that, while the connections are a relatively small percentage of the total steel mass

of a building, their cost is a major component of the overall economy of structure. Connections are repetitive by the nature and any savings in materials and labor can have significant effect on the overall economy of the building (Davison & Owens, 2011).

1.2 Research objectives and methods

The main objective of this thesis was to implement a toolkit that automates the steel connections design in BIM models by applying machine learning techniques. This study seeks possibility to assist design process by analyzing structural member attributes and geometrical relationships between them in earlier modeled buildings and then by using this information model connections in future projects.

Investigating the connection design process, BIM and machine learning formed an important part of the research. After a comprehensive literature review, the needed elements of the toolkit were clear and developing the toolkit was started. The toolkit was tested on 13 steel structures with the goal to find out do models have similar connections and is it possible to automatically model the connections.

1.3 Structure of the thesis

The thesis is divided into four sections; introduction (section 1), literature reviews (section 2), research approach (section 3) and results (section 4). The first section explains the motivation and the background of the subject and describes the objectives of this thesis. Additionally, the first section specifies the structure of this thesis.

The second section presents theoretical background of the thesis and literature reviews on three subjects: the theory of connection design, BIM and machine learning. The literature review on the theory of connection design explains basic of the connection design process and concepts acknowledged when choosing connections. The BIM literature review examines the development of BIM and the benefits and barriers of implementing it. In the machine learning literature review is explained the basics of machine learning and how it has been researched in BIM.

The research approach is presented in the third section. The section describes constructive part of the thesis and presents the chosen implementation of the toolkit. The section explains how machine learning and BIM software were made to collaborate, and it describes the structure of the database that was used for finding and modelling connections.

The fourth, final section summarizes the findings of the research work. The fourth section introduces the results from the case study and evaluates how the prototype application succeeded and what is there to improve. In the fourth section, a conclusion is provided, and further research areas are suggested.

2. THEORETICAL BACKGROUND

In this chapter, the principles of steel connection designing, BIM, and machine learning are presented. The chapter starts with introducing steel connections and criteriums how they are selected. This is followed by investigation of BIM and the way data is stored in BIM. Finally, machine learning and how it could be adopted in BIM is presented.

2.1 Introduction to steel connections

The word steel connection refers to multiple steel component which mechanically fasten the members. The components, including plates, welds and bolts, are used to join individual structural members of a steel structure allowing the structure to behave as intended. An example connection can be seen in Figure 2.

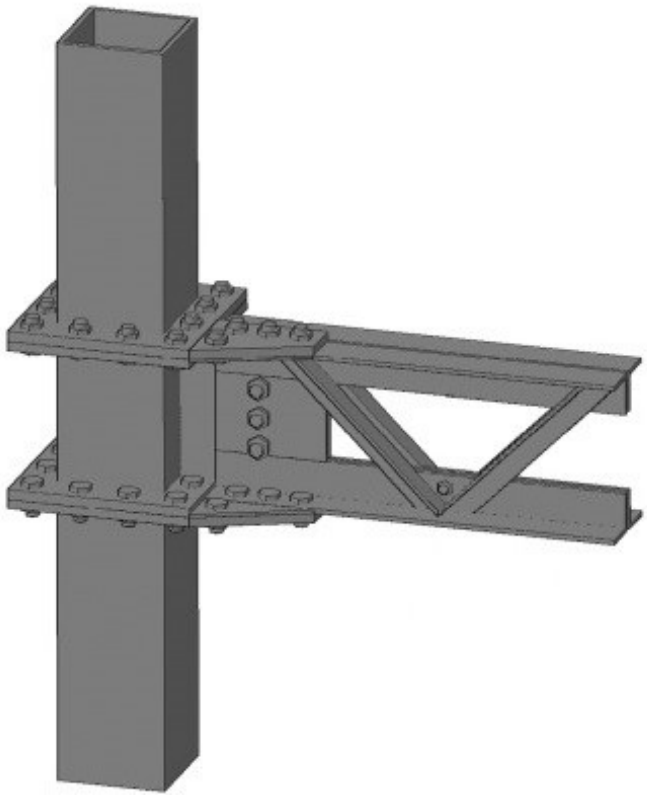

Figure 2. *An example of assembled connection (adapted from Liu et al., 2015).*

Normally, a steel structure project contains between hundreds to thousands of connections. While the connections are a relatively small percentage of the total steel mass of a building, their cost is a major component of the overall economy of structure (Corenc

et al., 2015) and the majority of the fabrication cost are generated by the connections (Davison & Owens, 2011). Moreover, choices of connections have significant influence on the cost of erection by affecting erection speed and easiness (Davison & Owens, 2011). This cost is, to a great extent, generated in the design phase.

Connections can be classified multiple different ways, but classifications based on the stiffness of the connection and on its resistance are two of the most significant (Rugarli, 2018). The connection types are usually set into three categories based on the stiffness of the connection: rigid, semi-rigid and simple. Even though classifications partly lose their usefulness when the connections are analyzed from 3D perspective, a one member force classification is a convenient indicator (Rugarli, 2018).

2.1.1 Components in steel connections

Structural connection is a complex entity and number of components in it can variate from few components to tens of components. These components work together like the links in a chain and like in a chain, the weakest component controls the strength of the connection. Component types can be divided into four categories: bolts and single-point type fasteners, welds, components such as connecting plates, gussets, cleats, and brackets, and members at the connection (Corenc et al., 2005).

Bolts and welds in steel construction are used connect components and members together. In general, site connections are usually bolted for the speed of erection and shop connections welded, but in special cases bolts are used in shop connections and welds are used in site connections (Corenc et al. 2005).

Bolts are used in several types of connections and can be used in all types of frame. The reasons for the popularity of bolted connections are (Corenc et al., 2005):

1. Low sensitivity to dimensional inaccuracies in fabrication, shop detailing or documentation
2. Simplicity and speed of installation
3. Low demand on skills of workers
4. Relatively light and portable tools.

Bolts in steel construction are categories to three categories: commercial bots, high-strength structural bolts, and precision bolts (Corenc et al. 2005). Commercial bolts are the most used bolts in steel construction, where high-strength structural bolts are used more demanding situation and precision bolts are commonly used as fitted bolts.

The main use of welds is in the production of steelwork and is particularly efficient useful for combining several plates and sections for increase of capacity (Corenc et al. 2005). Davison & Owens (2011) list following advantages of welding:

1. Freedom of design, and opportunity to develop innovative structures.
2. Easy introduction of stiffening elements.
3. Less weight than in bolted joints because fewer plates are required.
4. Welded joints allow increase usable space in a structure.
5. Protection against the effects of fire and corrosion are easier and more effective.

From these, the freedom of design is the main benefit of welded construction compared with bolted joints and welds enable some types of structures, such as tubular frames (Davison & Owens, 2011).

Welded joints may be divided into five groups: butt splices, lap splices, T-joints, cruciform, and corner joints. And for each of these groups there is a choice of three main types of welds: butt, fillet, or compound. These groups have their pros and cons, such as butt joints are preferable from purely strength considerations but preparing the plates for welding makes them relatively costly, and, in contrast, fillet joints require only minimal weld preparations and faster to execute, therefore less costly, but they do alter the flow of stress trajectories (Corenc et al. 2005).

Generally, structural members are not connected directly to each other with bolts and welds, but other steel components are needed. Naturally, these components are needed to transfer forces from member to another but also, they are used to accelerate erections by making it easier to operate with bolts and welds. Further, to reduce the cost of connection, forces are transferred as direct and simple way as possible (Corenc et al. 2005).

2.1.2 Connection choosing process

Designing connections are labor-intensive work. As mentioned before, a connection contains multiple components and adjusting one component may require modifications in others as well. Further, designers have often multiple options for the connection types and choices are different criteriums (Davison & Owens, 2011). For example, as can be seen in Figure 3, beam-to-column connection has multiple options of different types of connections and each of them have their advantages and disadvantages. A vast range of suitable connection types makes choosing most appropriate ones for designs more difficult (Davison & Owens, 2011).

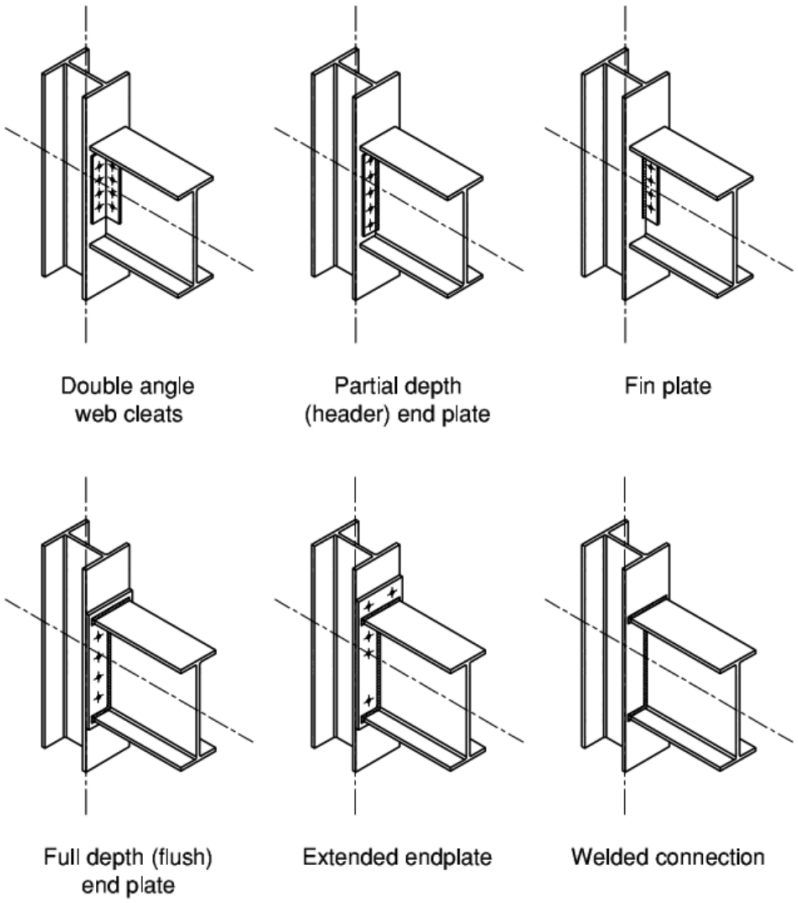

Figure 3. *Typical beam-to-column connections (Davison & Owens, 2011).*

The whole process takes multiple steps. For that reason, Corenc et al. (2005) introduces the most important principles to keep on mind while designing details:

1. Design for strength
2. Design for fatigue resistance
3. Design for serviceability
4. Design for economy.

Design for strength is the most crucial part for functional and reliable connection. As a purpose of a connection is to transfer the loads from member to another, it is clear that it cannot be done if the connection does not have enough capacities. Design for strength includes three points to be considered: direct force-transfer path, avoidance of stress concentrations, and adequate capacity to transfer the forces involved (Corenc et al. 2005).

Design for fatigue resistance is an addition to design for strength. Fatigue stress is a term for load cycles where higher and lower stress repeat in cycles. This can lead into fatigue damage and later into fatigue fracture (Corenc et al. 2005). Design for fatigue resistance has two major elements: avoidance of notches and careful design of welded joints.

For designing for serviceability, avoidance of features that can cause collection of water, ease of application of protective coatings, and absence of yielding under working load are the most important principles (Corenc et al. 2005). All of these are related to ensuring a long life span as well as making the installation to be as easy as possible.

While long life span and easy installation are necessary for economic connection, design for economy includes following three criterions: simplicity, minimum number of elements in the connection and reducing the number of members meeting at the connection (Corenc et al. 2005). Benefits to economy of these criterions can be seen clearly. Simplicate includes two parts; firstly, simpler connections use less elements and secondly, manufacturing complex elements is naturally more expensive than manufacturing simple, less work needed elements. Minimum number of elements in the connection and reducing the number of members meeting at the connection is related to simplicity. The goal of both of these criterions is to make connections as simple as possible without sacrificing other principles.

2.1.3 Connection capacity and verification

As above discussed, connections are essential to ensure that the outcome is a reliable building (Corenc et al., 2005). Yet, according to Davison and Owens (2011) detailing is often regarded as being of secondary importance in the designing process. Nonetheless, this may be changing. For example, Eurocode 3 pays greater attention to connection design than any code or standard before and an entire part, EN 1993-1-8, is dedicated to the connection design (Davison & Owens, 2011).

Connections capacities are assembled from multiple different properties. Davison and Owens (2011) provides three fundamental properties for connections capacities, that can be used to classify connections:

1. Moment resistance: connection may be either full strength, partial strength, or nominally pinned, in other words not moment-resisting.
2. Rotational stiffness: connection may be rigid, semi-rigid, or nominally pinned, in other words no rotational stiffness.
3. Rotational capacities: connections may need to be ductile. In the other words, a connection may need to rotate plastically at some stage of the loading cycle without failure.

While EN 1993-1-8 (cited in Davison & Owens, 2011) makes it possible to classify connections with different criteriums, it also gives three possible connection models based on the different frame analysis approaches:

1. Simple: connections are assumed to transmit no bending moments.
2. Continuous: connections are assumed to have no effect on the analysis.
3. Semi-continuous: connection needs to be taken into account in the analysis.

Based on these design methods, connections can be divided into two groups: simple connections and moment connections (Davison & Owens, 2011). Simple connections are defined as those connections that do not transmit moments at the ultimate limitation state and therefore they transfer end shear forces only (Davison & Owens, 2011). However, in reality, the connections do have some resistance to rotation. While it cannot be considered in the design, it is often enough to allow erect without temporary bracing (Davison & Owens, 2011).

Moment connections do transmit moments as the name suggest. This makes moment connections more complex in their behavior and the distribution of stresses and forces within the connection depends on the both the capacity of the components, such as welds and bolts, and on the relatively ductility of connected parts (Davison & Owens, 2011). Therefore, when choosing and designing connection. it is not enough to take into account only moment and shear resistance, but also stiffness of the connection and rotational capacity (Davison & Owens, 2011).

2.2 Building information modelling

BIM is defined in many different ways and the definition includes different elements that variate from person to another. BIM can be viewed from two windows: technical and philosophical. The technical view highlights ability to create and manage databases and preserve information for reuse. On the other hand, BIM is a philosophical framework that offers change and possibilities in construction industry.

In effect, BIM is both of these definitions and everything that comes between them and it is used from managing information to improving understanding (Dastbaz et al, 2017). The National Building Information Modeling Standard (NBIMS) (cited in Eastman et al., 2011) categorizes BIM three ways:

1. as a product
2. as an IT-enabled, open standards-based deliverable, and a collaborative process
3. as a facility lifecycle management requirement.

While the benefits of BIM in designing process, such as earlier and more accurate visualization of designs and reduced time for corrections when changes are made to designs, are clear, development of BIM tools have been slower than expected (Eastman et al, 2011).

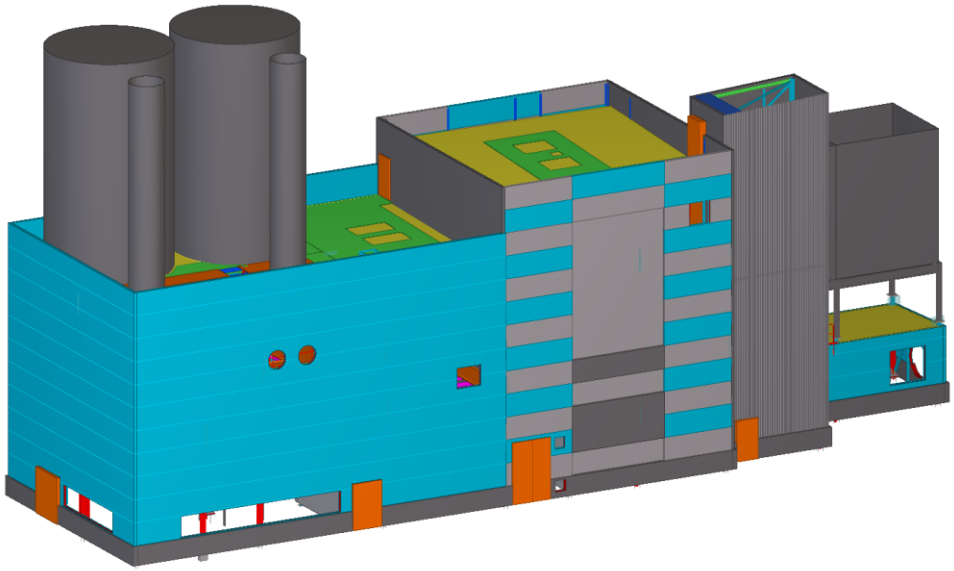

Figure 4. *An example of BIM model in Tekla Structures.*

Since BIM model have multiple purposes, generating them is a time-consuming task. While the tools offer ways to quickly modify the models, the goal of generated models is to be exact matches to buildings to be realized and achieving that can be time-consuming (Dastbaz et al, 2017). An example of a building created in BIM software can be seen in Figure 4.

2.2.1 Detailing in BIM

The design phase normally accounts for 5-10% of the total cost of a project (Yarmohammadi, 2017). Design phase at structural designing perspective can be divided into two phases: a conceptual phase and a detailing phase, where a detailing phase usually follows a conceptual designing phase (Chi et al., 2015). Generally, the goal of conceptual designing phase is to generate a basic structure of buildings while regarding requirements, such as owners' demands, structural codes and aesthetics (Chi et al., 2015). Detailing phase includes multiple tasks for accomplishing an exact digital representation of a building, connection designing being one of them.

It has been said that joint design is a bottleneck of structural design and it has large potential for improving the design process (Heinisuo et al., 2010). However, lack of flexibility, monotonous or time-consuming labor works, and communication gaps with different design aspects are stalling the development of structural design (Chi et al., 2015). Moreover, design solutions at the early stages may be limiting the modelling process at

detailing phase and affecting to opportunities to find superior design solutions (Chi et al., 2015).

Typically, connections contain multiple components (Corenc et al., 2005), as can be seen in Figure 5, and it makes their modelling complex and time consuming. Moreover, components are more complex geometrically than a structural member (Owens & Cheal, 1989). While this complexity affects on connection behavior under loads (Owens & Cheal, 1989), it also extends time it takes to model them. Components have multiple properties and attributes that have to be accounted for but, often, the BIM softwares includes tools that allow engineers to create the structural connections faster than creating them by using primitive BIM objects. This speed increase is due to the fact that these tools automate part of the normally manual work of the designer.

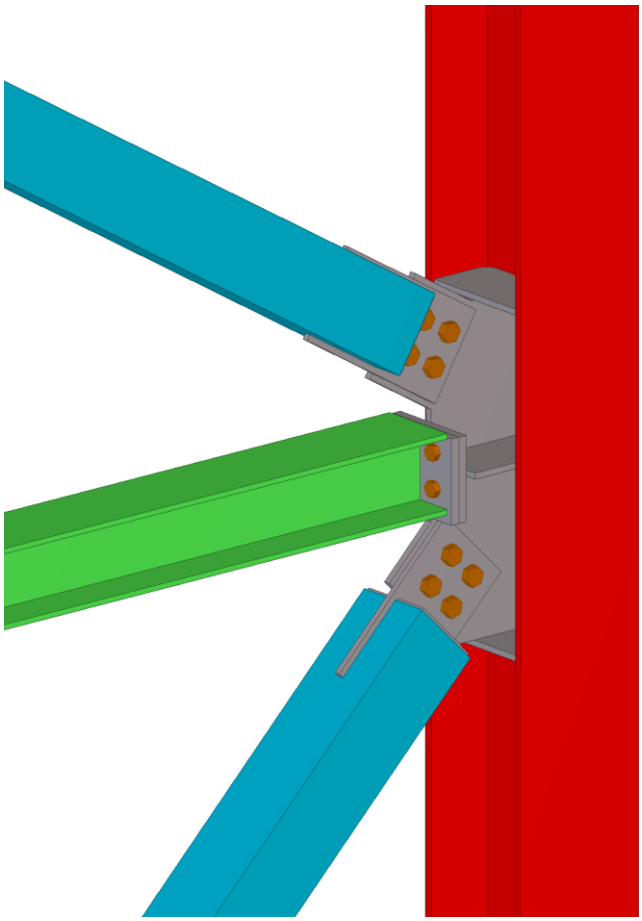

Figure 5. *An example of a connection in a BIM model.*

These tools allow designers to create a specific structural connection in a specific situation. Typically, the tools have premade rules, based on for example on specific manufacturers design guides, on which they create all the necessary components for a connection. However, the tools usually have numerously attributes and changing those attributes can be time-consuming. On the other hand, with these tools, editing a connection

is faster than editing a connection created manually. Moreover, these tools allow engineers to save the used values which allows creating similar connections faster and with less unique components in connections.

However, for example by using global IFC standard, it is possible to collaborate between modelling softwares that have different functionalities. This expands possibilities to design and model connections. In this way, by supporting use of third party softwares, connection designing and calculating can done in a specialized software and designed connections seamlessly brought to a BIM modelling software.

On the opposite side, designers are quite often obligated to design connections without using these semi-automated tools. Modelling connections part by part is an extensive task and the modelled connections cannot usually be translated to be used later. This process takes considerable time because number of the parts and time it takes model a single part. While it is possible to develop parametric tools for each connection type, some connections are quite unique and used rarely. In these cases, generating tools for them is not cost-efficient and time spent in modelling cannot be utilized somewhere else.

2.2.2 Storing data in BIM and in CAD

While CAD provided new tools, and supported new ways of working, CAD adoption was mostly based on substitution of existing practice modes (Kensek & Noble, 2014). The decisions to start use CAD usually based on discussions of drafting speed, ease of making updates, and the limited benefits of enhanced accuracy (Kensek & Noble, 2014) as the original concept of CAD was to be able draw simple lines quickly and easily without having to draw on paper (Weygant, 2011). Naturally, when CAD technology grew, it allowed more advanced actions, such as categorizing lines into various layers (Weygant, 2011).

The difference between CAD and BIM is not just a difference between 2D and 3D designs since CAD can also offer 3D representation, but also amount of the information (Dastbaz et al., 2017). While, in CAD, a building and its elements are represented by lines and other geometrical shapes, in BIM the elements contain more detailed information, such as material (Dastbaz et al., 2017). Shift from CAD to BIM gave designers ability to look at not just what an element looks like, but what it is (Weygant, 2011). An element, that was represented in CAD as a series of lines in different views, is an object with its properties and attributes. Weygant (2011) demonstrates this by simple saying BIM is basically CAD with specifications and emphasizes that the specification is where the real value of BIM lies.

The increase of the data has changed the way data is stored. In CAD designs, there is not that much data for need to implement a complex database. Since the stored information is only graphic information, there are only few ways to organize the different objects (Weygant, 2011). However, in BIM models, where a single object contains numerous different attributes, the databases are much more sophisticated. It is more natural to have hierarchical database since the content naturally has hierarchy. Objects usually evolve based on the amount of information known about it (Weygant, 2011). For instance, a window object might inherit the properties and attributes an opening has, but an aluminum casement window has some extra properties and attributes compared to a window.

Moreover, the content in BIM models can be usually divided into objects and assemblies (Weygant, 2011). Assemblies are a concept of series of objects that work together to create a single element, such as a wall or stairs. Assemblies contains not only the information objects have but possible its own information. Since assemblies may contain other assemblies inside it, assemblies add hierarchy into databases (Weygant, 2011). In Figure 6 is an example of hierarchy that demonstrates how material interrelates with objects and assemblies.

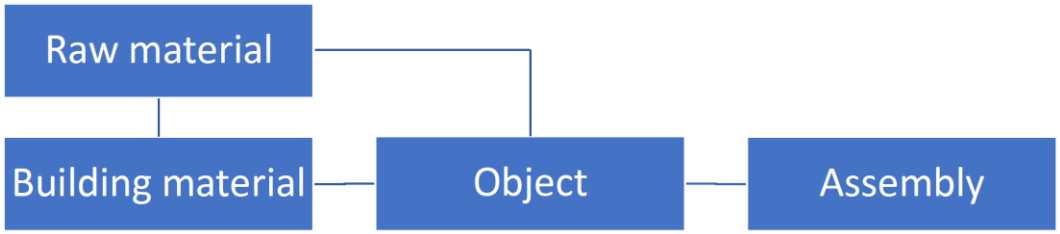

Figure 6. *Simplified interrelation between material, object and assembly (adapted from Weygant, 2011).*

According to Weygant (2011), BIM models are databases with graphical user interfaces with small additions. The databases contain everything put into models and they should contain enough information about objects and assemblies that they can accurately specify actual products (Weygant, 2011). Weygant (2011) categorizes the information associated with objects into five categories:

1. Identification; What is the product?
2. Performance; How does the product work?
3. Installation; How is the product installed?
4. Appearance; What does the product look like?

5. Lifecycle and sustainability; How is this product maintained?

In BIM models, the information is stored as parameters and attributes. These parameters and attributes are series of pairs that contain a name and a value (Weygant, 2011). While dimensions and material are the most important, as they determine the overall appearance of the component that is being developed, there are several types of attributes and parameters that may be used in BIM (Weygant, 2011). Weygant (2011) separates the types into eight most commonly used groups: length, area, angle, text, boolean, number, integer and hyperlinks. However, by approaching the types from programming perspective, they can be divided into 3 three categories: integers, real numbers, and strings.

2.3 Machine learning

Machine learning is a branch of artificial intelligence and a technique of learning from data. Based on the wanted outcome, information and trained machine learning models can be used, for example, to generate a prediction. It is widely used in many fields, ranging from Computer Science, to Physics and Biology. These fields have different kind of data to feed into algorithms and wished output variates between cases. Flach (2012) believes that the diversity of input and output options and the ubiquity of the tasks that can be solved by machine learning are helping to make machine learning powerful tool for virtually every branch of science and engineering. The diversity provides numerously machine learning algorithms that can be chosen based on required output (Bell, 2014).

Typically, machine learning algorithms fall into one of two learning types: supervised and unsupervised learning (Bell, 2014). Supervised learning includes algorithms that work with labeled training data. On the opposite side is unsupervised learning, where algorithms find the patterns or labels from data.

In supervised learning, for every example in the training data an input object and output object are needed (Bell, 2014). Typically, based on the training data, supervised learning algorithms try to find rules how to map input objects to output objects. However, while predictive models are the most common setting, supervised learning algorithms include descriptive models that are not primarily intended to predict the output object, but identifies differently behaving subsets of data (Flach, 2012). Validating the results is often easy and it is typically done by dividing the training dataset into a training dataset and into test dataset and comparing how well the algorithm is predicting correct output object (Flach, 2012). However, supervised learning has issues to be considered, such as the outcome result is often needed to be added manually and bias-variance dilemma (Bell, 2014).

Unsupervised learning can be considered to be more datamining than actual learning from data (Bell, 2014). The goal is to discover hidden patterns in the data (Murphy, 2012) and a typical example of unsupervised learning is to cluster data with intention to assign class labels to existing data (Flach, 2012). However, unsupervised learning algorithms include both descripted and predictive settings. Validating the results is harder than with supervised learning algorithms because there is no test data as such (Flach, 2012). It has even said that there are no right and wrong answers in unsupervised learning (Bell, 2014).

In addition to learning type categories, machine learning algorithms can be grouped by what is learned from the data and Flach (2012) groups algorithms into three groups: geometrical algorithms, probabilistic algorithms, and logical algorithms. However, Flach (2012) points out that, while these groupings are not mutually exclusive, they provide a good starting point.

For geometrical algorithms, a distance is essential concept (Flach, 2012). If the distance between two input objects is small then the input objects are similar and they probably belong to same cluster or get same classification. The distance can be calculated many ways, but one of the most used one is Euclidian distance. Euclidian distance can be presented in the following way:

$$d = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}, \quad (1)$$

where *d* is the Euclidian distance and x_i and y_i are distances along a coordinate. The distance can be used for example to classify input objects. As simplest, distance based classifier, nearest-neighbor classifier, classifies an input object by the shortest distance to training instance and applies its class to the object.

Probabilistic algorithms try to find a way to a relationship between input values and target values. The algorithms are using the data to find out an unknown probability distribution which is caused by a hidden random process that sets up the target values from the input values (Flach, 2012). For example, a probability that an e-mail is spam can be estimated by what words it contains. E-mails with words ‘casino’ or ‘lottery’ will be more likely considered as spam than the e-mails without them.

Logical algorithms are more algorithmic in nature (Flach, 2012). Algorithms generates rules from the data that can be translated into chain of if-then statements. The chain of if-then statements can be visualized in a tree structure that can be seen in Figure 7. The logical algorithms are usually decision trees, that are used to predict the output based

on if-then statements. While the decision tree looks a simple concept, Bell (2014) reminds that within their simplicity lies their power, such as they are easy to read and they perform well with reasonable amounts of data.

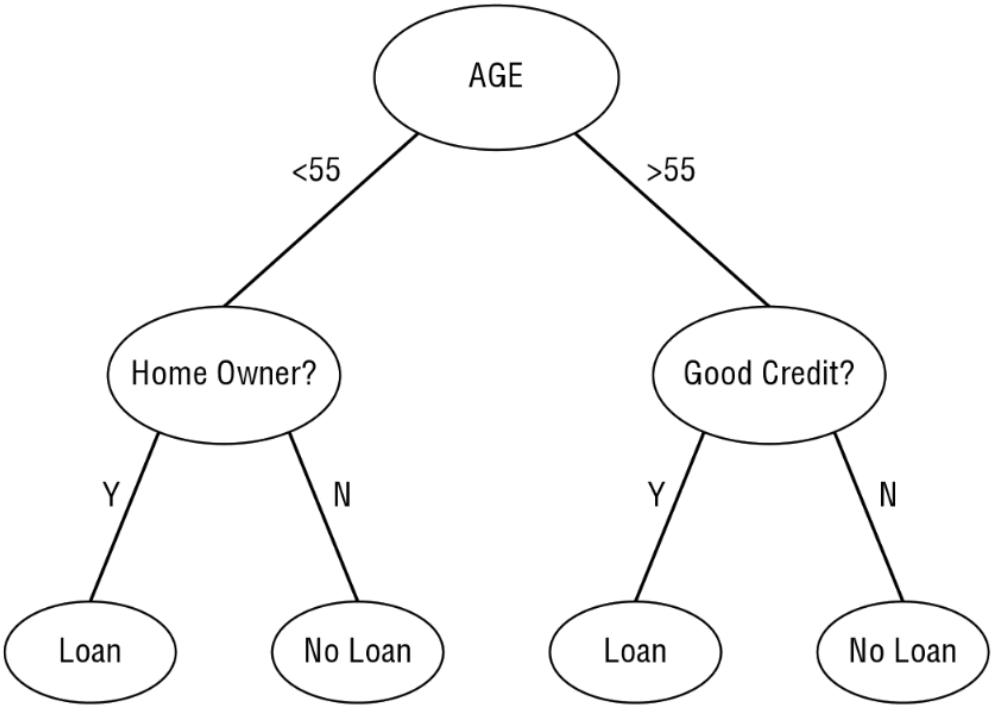

Figure 7. *An example of decision tree (Bell, 2014).*

2.3.1 Data types in machine learning

Machine learning can be applied to various of different fields, which makes data to differ by nature. Even in a single data set the features can be of different types and they can have different properties. The difference of feature types may cause problems when the data is fitted to model since algorithms are not often handling mixed data types well. The data types can be divided to four groups: nominal, binary, ordinal, and numeric (Han et al, 2011).

Nominal features are features that can be categorized, such as symbols or names of things, although the names can be represented with numbers (Han et al., 2011). For example, feature member type, such as column or beam, is a nominal feature. It cannot have any meaningful order and it is not quantitative. This means it is not possible to find average value or median value, but instead, the most common value can be defined.

Binary features are an extension to nominal features, where nominal features can have multiple states, binary features have only two categories or states: 0 or 1 (Han et al.

2011). Usually 0 means that feature is missing, and 1 means that it is present. For example, feature describing a connections assembly place whether it is assembled in a workshop or on a site, is a binary feature.

Ordinal features are the other extension to nominal features. The difference between ordinal features and nominal features is that in ordinal features, the values have meaningful order or ranking among them, but the magnitude between successive is not known (Han et al., 2011). For example, features material and profile are nominal values. These nominal features have multiple possible values, but they can have a meaningful order, such as strength or size.

Numerical features are quantitative and they are represented in integer or real values (Han et al., 2011). Numerical features can be divided into two categories: interval-scaled and ratio-scaled. Difference between interval-scaled and ratio-scaled values is that interval-scaled values can be positive, 0 or negative and don't have true zero point where ratio-scaled features have clear zero-point.

Data, in which more than one type of feature are present, is called mixed data. Generally, data consist several types of features as they all can bring insight for the problem. However, mixed data may cause problems since many machine learning algorithms are not handling well data with multiple types (Han et al, 2011). Luckily, there are several methods to transform the data to compatible format. For example, there is several methods for encoding nominal data to numerical format, of which assigning each value a number is perhaps the most intuitive way.

2.3.2 Clustering

Unsupervised learning includes multiple subcategories, clustering being one of them. Clustering in machine learning is an assignment to set data into subsets, using the fact that objects in the same cluster are more similar to each other than to those in other groups (Jain, 2010). Example of clustering can be seen in Figure 8.

According to Jain (2010), the target of clustering is "to discover the natural grouping(s) of set of patterns, points, or objects". However, the definition of clustering contains some loosely defined words and that's why it is hard to say when similarity ends or how dense clusters should be. For a human, seeking clusters in a two dimensional or a three dimensional data is not a problem, but with high-dimensional data automatic algorithms are necessary (Jain, 2010).

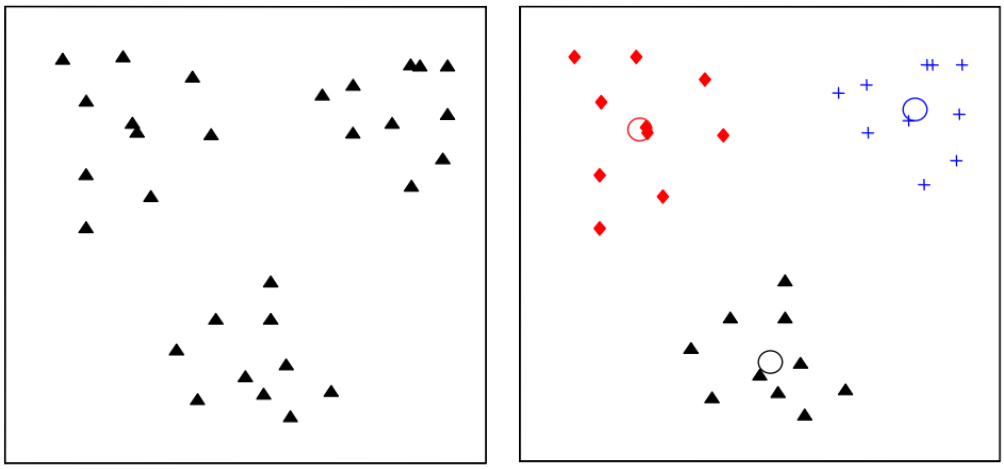

Figure 8. *Illustration of clustering. (Left) Two-dimensional input data; (Right) final clustering obtained by K-means algorithm. (Jain, 2010).*

Jain (2010) says that data clustering has been used mainly for following purposes: underlying structure, natural classification, and compression. Underlying structure includes purposes for gaining insight into data, generating hypotheses, detecting anomalies, and identifying salient features. Natural classification means identifying the degree of similarity among forms and organism and compression is used as method for organizing the data and summarizing it through cluster prototypes.

Because of that, clustering has been used widely in different fields and there is an enormous number of researches where it has already been used (Han et al, 2011). The differences of these use cases emphasize that clustering excels multiple areas and according to Jain (2010), clustering is currently a key strategy in disciplines involving the analysis of multivariate data.

However, even though clustering is widely used and it has been researched for decades, the number of clusters has been one of the most difficult problems in data clustering (Jain, 2010). Usually, the number of clusters *K* is predefined and chosen based on criterion created by domain knowledge. However, there is approaches to automatically estimate *K*, but deciding which value of *K* leads to more meaningful clusters is a complicated task (Jain, 2010).

2.3.3 Classification

The main task of classification is to predict class labels based on a training dataset (Flach, 2012). The algorithm processes a training set containing a set of attributes and the respective outcome in order to predict the outcome by discovering relationships between the attributes (Voznika & Viana, 2007). When the algorithm is given a test dataset

not seen before, which contains same set of attributes but missing the labels, it will analyze the input and predict the outcome. Figure 9 is an example that demonstrates the learning and classification processes for predicting loan decision.

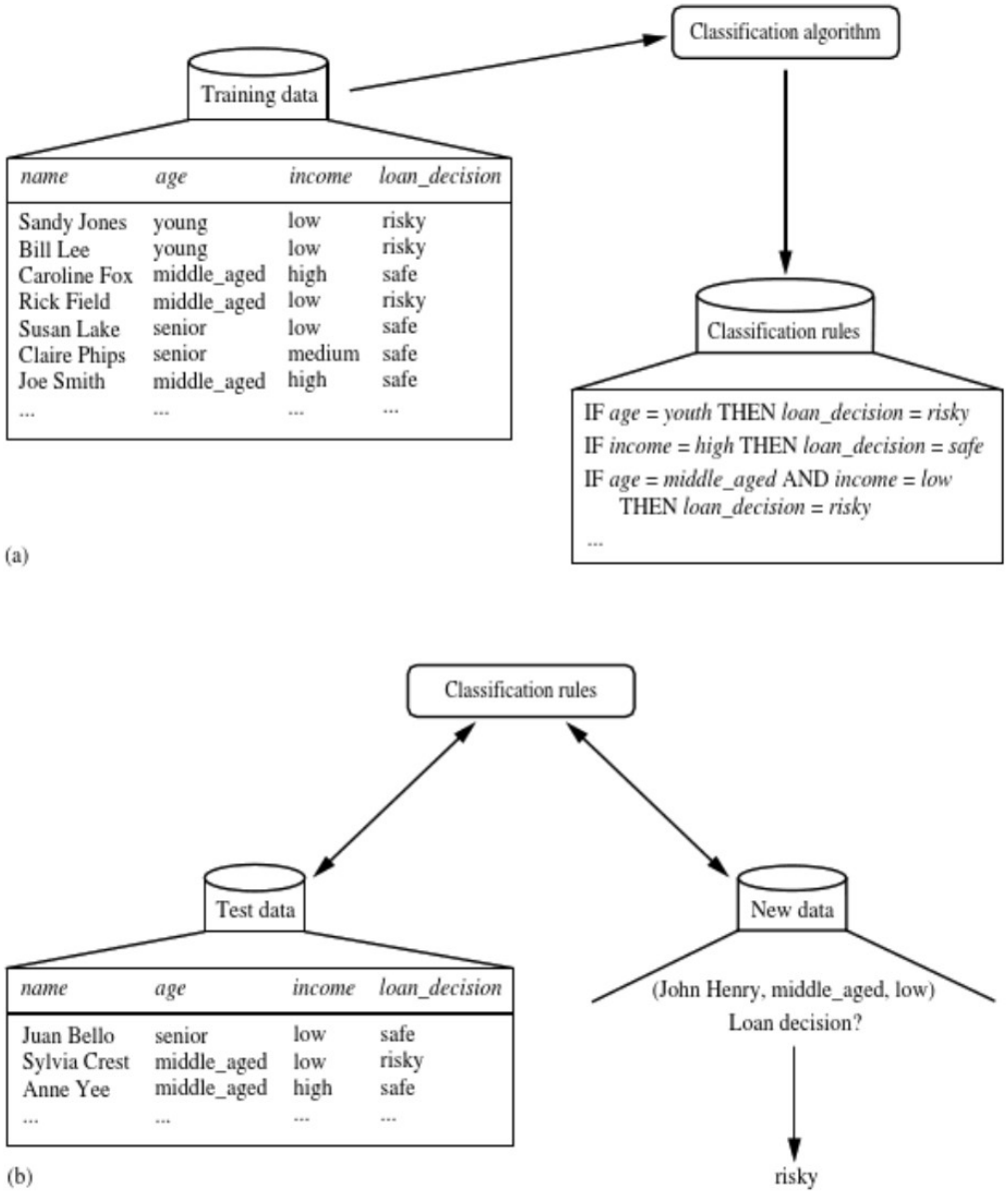

Figure 9. *An example of classification process: (a) learning and (b) classification. (Han et al., 2006)*

Classification problems can be divided into two groups by number of classes: binary classification and multiclass classification. Flach (2012) says classification is the most common task in machine learning. It has also been said that classification is one of the most important research topics in data mining (Zhang et al., 2017).

Many classification methods have been developed over the past few decades. One of the most popular approach among them is the nearest neighbor (NN) approach (Zheng

et al., 2004). NN methods are instance-based learning. Instance-based learning is lazy since the real work is done when the time comes to classify a new instance rather than when training set is processed (Witten et al., 2011). Moreover, in NN methods, each new instance is compared with existing ones using a distance metric and closest existing instance is used to assign the class to the new one (Witten et al., 2011). According to Shakhnarovich et al. (2006), the NN approach is specifically appealing when searching the best match, if large amounts of data are available.

Shakhnarovich et al. (2006) define the NN problem in Euclidian space as “given a set P of points in a d-dimensional space Rd, construct a data structure, which given any query point q finds the point in P with smallest distance to q”. Moreover, Shakhnarovich et al. (2006) point out that defining the distance between a pair of points p and q is required to completely specify the problem.

To make the NN approach excel, large amount of data needs to available. In general, this is a problem when analyses are performed using machine learning (Chervonenkis, 2011). The lack of training data prevents validating results and increases risks related to choosing algorithms and training machine learning models. Moreover, Chervonenkis (2011) points out that the chance to find a good model within a small class is less than for a large class.

The quality of predictions is measured in percentage of predictions hit against the total number of predictions (Voznika & Viana, 2007). This is usually done by having a separate test dataset that can be used for testing. In Figure 9 (b), test data are used to estimate the accuracy of the classification rules.

2.3.4 Artificial intelligence in BIM

By definition, the artificial intelligence means computer programs performing tasks requiring intelligence when done by human (Butterfiled & Ngondi, 2018). This requires not only knowledge about the physical world, but with common sense knowledge (Davis & Marcus, 2015). In this case, common sense knowledge is referred to as the general knowledge about the world. For instance, if a person is holding a baby in his arms, and it is known that they are father and son, it is clear which is which. This type of knowledge is natural for human being and it is used for tasks like language processing. However, it is mostly missing from BIM softwares (Bloch & Sacks, 2018).

Intelligence in existing BIM softwares is commonly limited into parametric modelling and design intent behavior that maintains design integrity (Sacks et al, 2004). This means that we are far from being able to refer them as being intelligent (Bloch & Sacks, 2018).

While the models consist of elements that are of their properties and their relationships to other elements, they still allow users to take actions that are clearly not rational in the context of building design. This kind of rationality of actions is still missing from BIM models (Bloch & Sacks, 2018). As object-orientated system, BIM tools are mostly focusing representing objects, including their properties and relationships between objects.

3. IMPLEMENTATION

The purpose of this thesis was to implement a toolkit for automating the generation of steel connections of BIM with the help of machine learning. While none of the detailing or modelling softwares are using machine learning for automation and solutions to automate designing are very limited (Bloch & Sacks, 2018), this thesis also aimed to investigate if machine learning could be used within BIM. Although the capacity calculation of those modelled connections is left out of scope of this thesis, the toolkit estimates the capacities by comparing stresses of structural members.

The mechanic behind user interface (UI), called the core, can be divided into three modules: BIM handler module, database module, and machine learning module. The simplified structure of the modules and their main tasks can be seen in Figure 10.

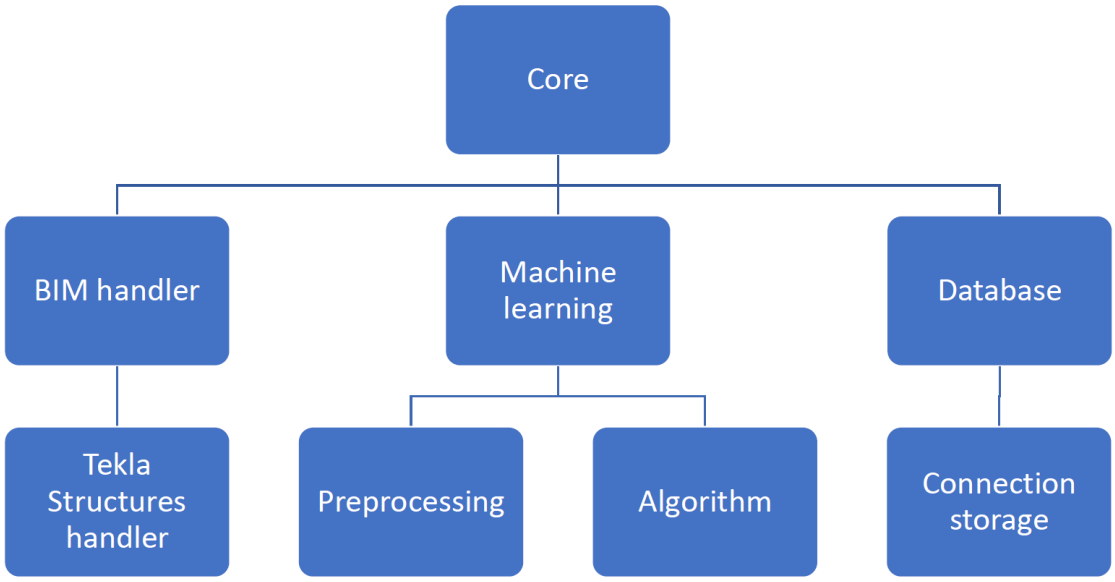

Figure 10. *Architecture of the toolkit.*

The BIM handler module is responsible for reading and creating objects in BIM models. It was designed to be as non-software-specific as possible to make possible to use the same toolkit with other modelling softwares. The toolkit was built to operate using the Tekla Structures software and other modelling softwares are left out of scope of this thesis. However, extending this toolkit to other BIM softwares can be done rather easily by building another sub-module inside the BIM handler module.

The machine learning module is handling everything needed for running machine learning techniques, including functionalities for preprocessing data and running algorithms. Finally, the database module is needed for reading and storing the information other two modules are providing.

3.1 Tekla Structures

The Tekla Structures software was chosen to its convenient functionality for automation. It has open application programming interface (API) – Tekla Open API™. The Tekla Open API provides interface for third party applications to interact with the model and drawing objects in Tekla Structures (Tekla Open API, 2018).

The BIM handler module was written using C#. C# is mature, object-oriented programming language while being part of .NET Framework (Ky, 2013). This language was chosen because in the Tekla Open API environment this is the most commonly used language, even though it is possible to use VBA as well (Tekla Open API, 2018).

However, as described earlier, the structure of BIM handler was made to be as non-software specific as possible. To accomplish this, the BIM handler module includes a separate sub-module that works with Tekla Structures. It was created as its own separate module for easy replacement by other modules for other designing softwares.

3.1.1 Automation in Tekla Structures

Modelling often includes multiple repetitive tasks. To reduce that, BIM softwares have approaches to automate those tasks. Tekla Structures has a functionality to extend capability of modelling by using extensions which can be used to automate repetitive designing tasks. In Tekla Structures, there are five different ways (Figure 11) to create extensions: external .NET applications, plugins, macros, custom components, and remote procedure call (RPC) macros (Tekla Structures Glossary, 2018).

The first three interact with Tekla Structures by using Tekla Open API. However, even though plugins are separate dynamic link library (DLL) files, plugins have to be loaded inside Tekla Structures process and can't be modified while the process is running (Tekla Structures Glossary, 2018). Moreover, RPC macros are ignored in this case since RPC macros are internal ways to modify an application and, in Tekla Structures, they are included in a program configuration.

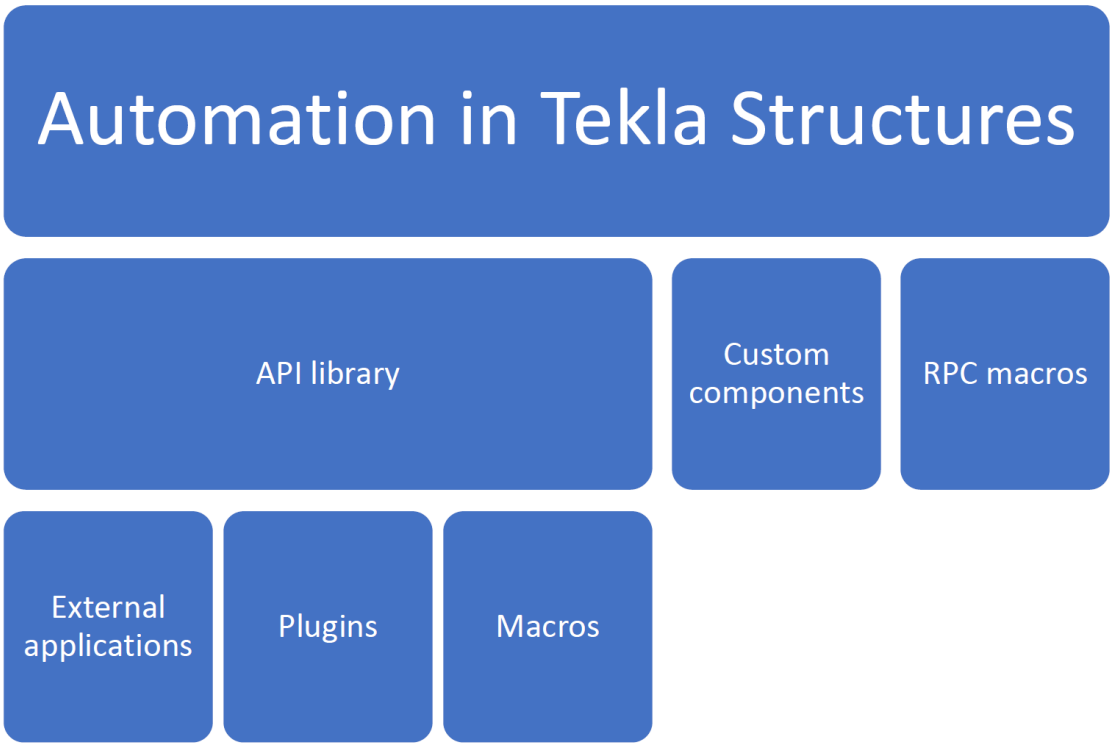

Figure 11. *Automation in Tekla Structures*

Custom components are a combination of different objects. Users can create them from model objects whose composition can be modified as a group. In this regard, they are very easy to create and use and they can be created without any programming experience using Tekla Structures tools. On the other hand, it is limited what they can do, and they are usually used for inserting simple combinations of objects.

By definition, macros are programmable patterns and they are used make tasks less repetitive. In Tekla Structures, macros are a saved series of actions that includes instructions for a program. They can be created by recording actions a user does or by writing the actions to a file. However, the commands are usually simple, and, in Tekla Structures, they are used to do simple tasks, such as exporting output file.

Plugins, on the other hand, handle better more complex structure of objects. Plugins modify models via components that are groups of model objects. They are easy to insert to a model and modify as a single unit. In comparison to custom components, the components have input fields for faster modification of objects (Figure 12) and they adapt to changes in the models. For example, a connection created by a component is automatically updated if the user modifies the parts it connects (Tekla Structures Glossary, 2018). However, plugins are usually made for a single use case, such as a single connection type. Moreover, plugins must be created using a high-level programming language, such

as C#, which requires software development skills which are not typically common among structural engineers.

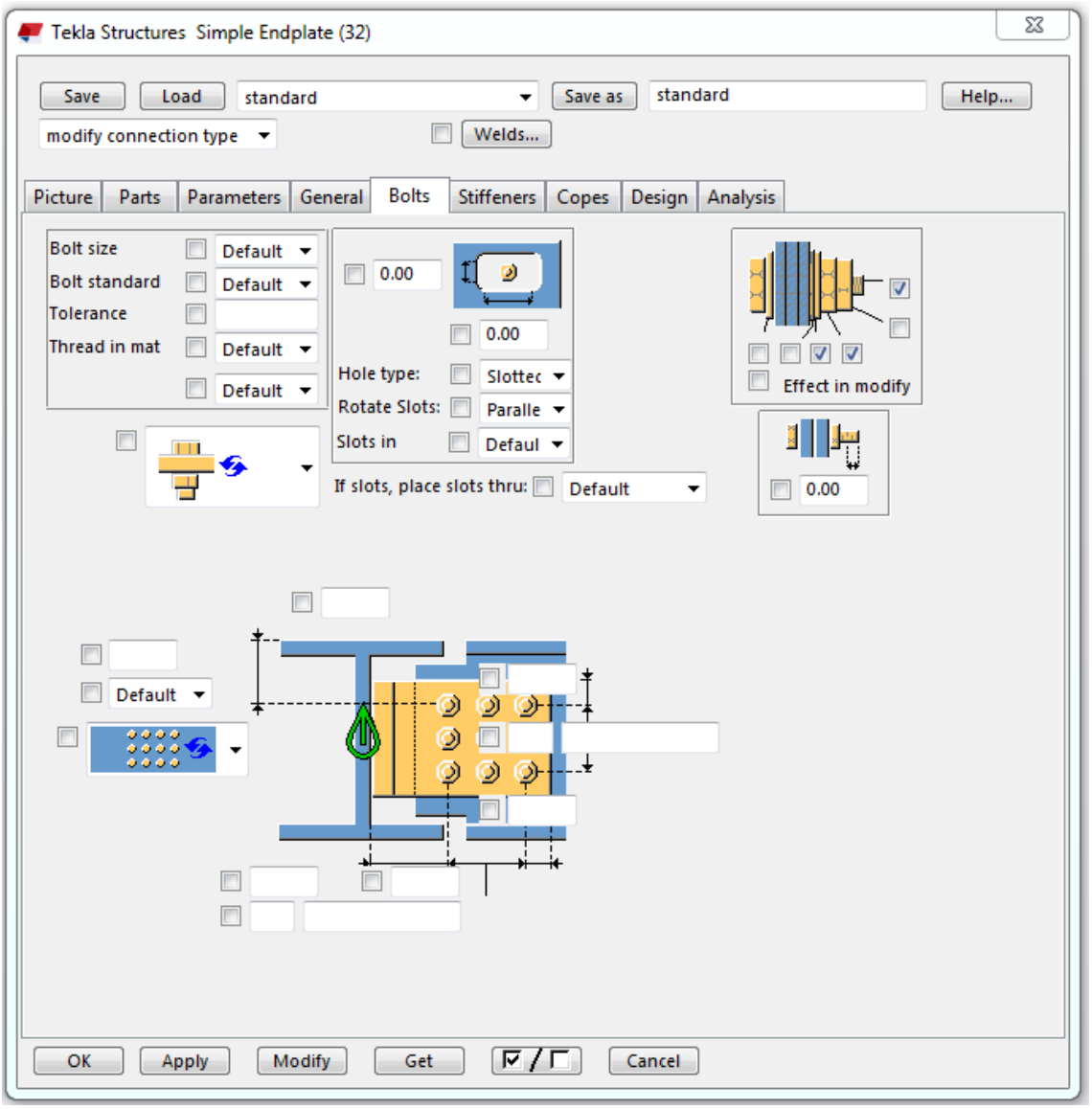

Figure 12. *An example of a plugin user interface*

The external applications are different way to approach this problem. In comparison to plugins, they don't have that much restrictions. They can be developed to be fully independent softwares that only connect to the Tekla Structures software when needed. This has raised new ideas how designing can be done. For example, parametric modelling and algorithms-aided design have been gaining more and more interest (Harding & Shephard, 2017; Lalla, 2017). The central idea of these methods is to modify models by changing input values by predefined rules instead of modifying the actual model. This increases modelling speed and improves adaptability to changes (Lalla, 2017). However, the main difference between parametric modelling and algorithms-aided designing is that

in parametric modelling, the model is modified by changing parameters, and in algorithms-aided designing, the model is modified by the outcome of algorithms (Lalla, 2017).

The rules needed for parametrical modelling and algorithm-aided designing raise the biggest obstacle of using parametric modelling and algorithms-aided designing. These methods need boundary conditions and thinking and implementing these conditions is a time-consuming task. For that reason, parametric designing is the most suitable for modelling geometrical complex structures and when it is expected that designs will change during process (Lalla, 2017). Moreover, Lalla (2017) says that modeling connections have more difficulties compared to structural members because they have considerably more parameters, and their geometry is often more complex.

3.1.2 Tekla Open API

As above told, Tekla Structures provides open API, Tekla Open API, which allows users to create their own plugins and external applications on top of Tekla Structures. The difference between these two implementations is that plugins are loaded while Tekla Structures is starting and run in the same process with Tekla Structures and external applications are run their own processes. However, external applications require Tekla to be open to be able to read and modify models (Tekla Structures Glossary, 2018).

Tekla Open API includes eight DLL files which can be seen in Figure 13. These libraries can be divided into three categories: user interface libraries, modelling libraries, and core libraries. The libraries provide basically all the same methods for editing models what Tekla Structures user interface provides.

UI libraries include tools for creating UI and other visual tools for plugins and external applications. Dialog library contains classes and methods for creating UI for plugins. Catalog library includes functionality to access Tekla Structures catalog instances, such as the profile or rebar catalog, but also UI components that can be used in a plugin UI.

Modelling libraries contain classes for handling structural parts in a model. Model library contains classes for structural parts and they are named similarly to ones in Tekla Structures UI. These classes include methods for creating, modifying and deleting them. Furthermore, the library provides tools for accessing to all objects in a model. Drawing library contains similar classes and methods for handling drawings and objects inside them. Finally, Analysis library provides basic classes that can be used for accessing analysis and design information.

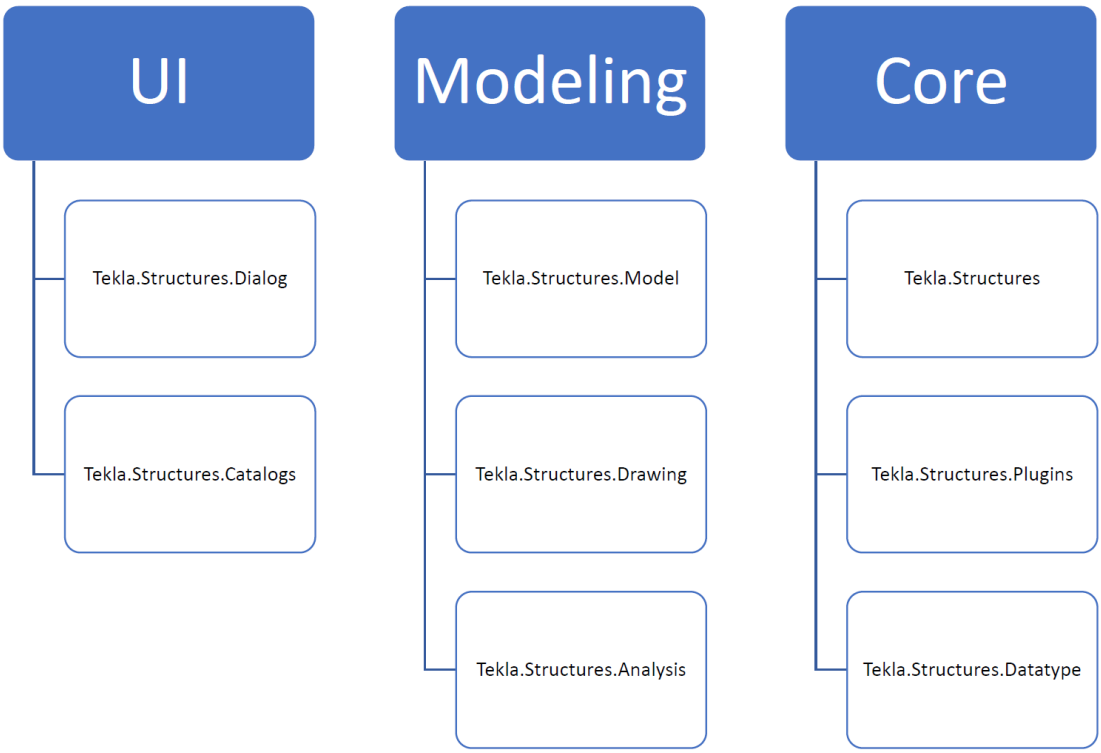

Figure 13. *Tekla Open API libraries*

Core libraries are used in other Tekla Open API libraries. Tekla.Structures library contains basic and common types that are shared between Model and Drawing libraries and methods to edit Tekla Structures settings and environment variables. Plugins library includes functionality to create plugins and abstract for classes which are needed to be inherited in plugins. Datatype library contains methods related to different datatypes which are used in other libraries.

Program 1 is an example of method written by C# that changes profiles of selected beams in Tekla Structures model for demonstrating Tekla Open API. The example contains three parts: getting selected beams, modifying those beams profiles and returning value if all beams were modified successfully.

For getting selected objects, a *ModelObjectSelector* object is created. By using a method from the object, all selected objects are got. Iterating through all the selected objects and casting objects to a *Beam* object is used for finding beams out of all selected objects.

If an object is a beam, the profile is changed to match the given profile string. If modification is not successful, information of that is stored variable named *beamsProfilesModifiedSuccessfully*.

Finally, if the all modifications are successful, true is returned. Otherwise false is returned. The reason for unsuccessful modification could be lost connection to Tekla Structures, for example from closing the program while the method is running, or a wrong value in profile string.

```
public bool ModifyProfileOfSelectedBeams(string profileString)
{
    bool beamsProfilesModifiedSuccessfully = true;

    // Object for finding selected model objects in Tekla Structures
    var modelObjectSelector = new Tekla.Structures.Model.UI.ModelObjectSelector();
    ModelObjectEnumerator selectedObjects = modelObjectSelector.GetSelectedObjects();

    // Iterates thorugh all the selected model objects
    foreach (ModelObject modelObject in selectedObjects)
    {
        // Cast model object to beam for filtering other object types
        Beam beam = modelObject as Beam;
        if (beam == null) continue;

        // Modifies beams profile
        beam.Profile.ProfileString = profileString;

        // Applies the modification
        var modificationResult = beam.Modify();
        if (modificationResult == false)
            beamsProfilesModifiedSuccessfully = false;
    }

    // Return true if profiles of all selected beams are modified successfully
    return beamsProfilesModifiedSuccessfully;
}
```

Program 1. *An example method of modifying selected beams profiles by using Tekla Open API*

3.2 Machine learning

A machine learning project contains typically 4 main actions: acquisition, prepare, process and report (Figure 14). These actions of implementing the machine learning module are explained in following chapters.

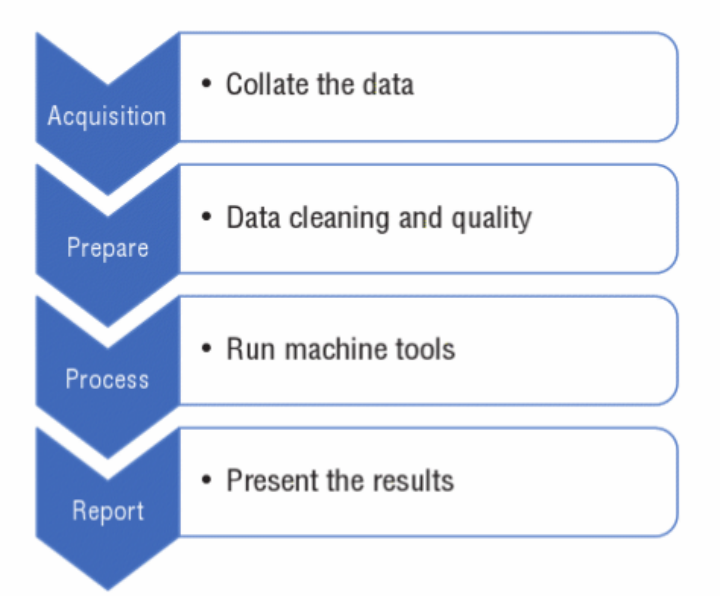

Figure 14. *Action in a machine learning project. (Bell, 2014)*

In the acquisition, information from connections were collected and stored to a database and a dataset was created. In preparation part, the dataset was preprocessed to a format that machine learning algorithms can use. The processing phase is where the algorithm is run and the actual work gets done. Finally, in the reporting, the results are presented to users.

Understanding inputs and outputs are more important in any software system, machine learning being no exception, than knowing what goes on in between (Witten et al., 2011). After understanding the input, connections in the database, and the output, prediction of suitable connections, it is easier to analyze suitable approaches and algorithms and evaluate needed features.

3.2.1 Creating database

As earlier described, the input fed into machine learning algorithm is connections obtained from past BIM models. Each connection in a model were added as a new instance containing data obtained from them. The acquisition of information was automated for better user experience. However, reading stresses from structural analysis software have to be done manually which naturally increases designers' tasks. The obtained information included two parts: information for finding the best matching connections and information for modelling those connections. The main structure of the database can be seen in Figure 15.

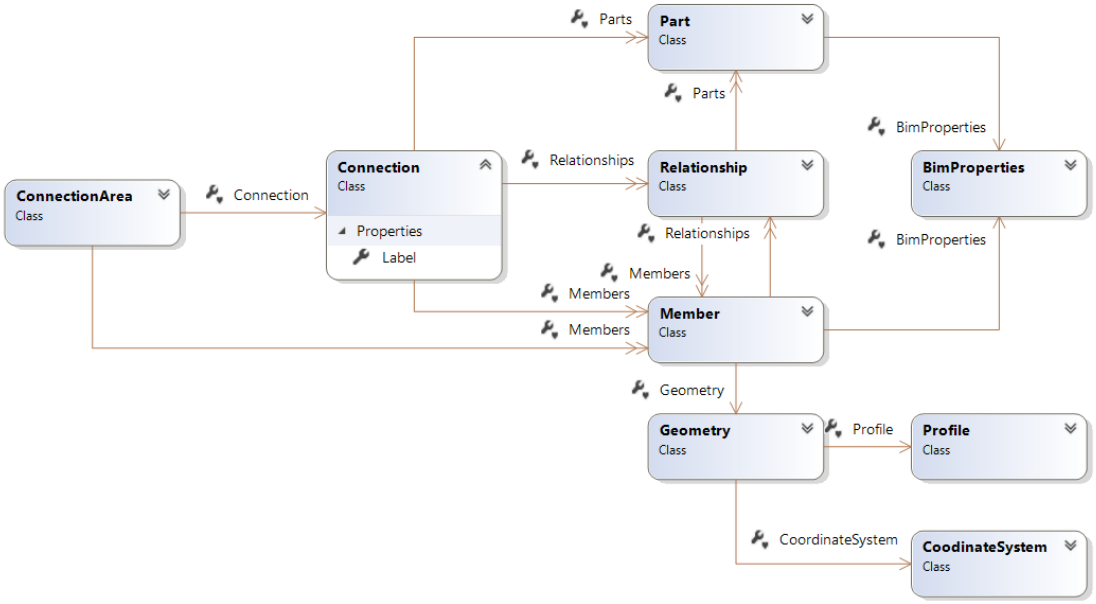

Figure 15. *The main structure of the database*

The information for finding connections have two major elements: geometrical relationships and BIM properties. Geometrical relationships include the relations between structural members including information such as positions and angles between members. BIM properties include properties obtained from BIM model such as material.

For remodeling a connection, the toolkit needs all properties of parts used in that connection. As the goal was that there is no difference between a user modelled connection and an automatically modelled connection, it was not possible to leave out any properties. However, creating identical parts is not enough, the parts must be same hierarchical structure as well as connected to other parts in the same way. The relationships between parts from connections and structural members were saved to the database.

In this study, a connection area is defined as a variable that contains information on the structural members linked by a given structural connection and on their geometrical relationship. Meanwhile, a structural connection is defined as a group of individual components that link structural members. An example of a connection area is shown in Figure 16.

However, connections did not contain information about labels and exact labelling is a time-consuming task. While connections can be separated to different types, more accurate labelling is hard to do. For example, a connection where number of bolts or plates thickness is changed is not the same connection anymore.

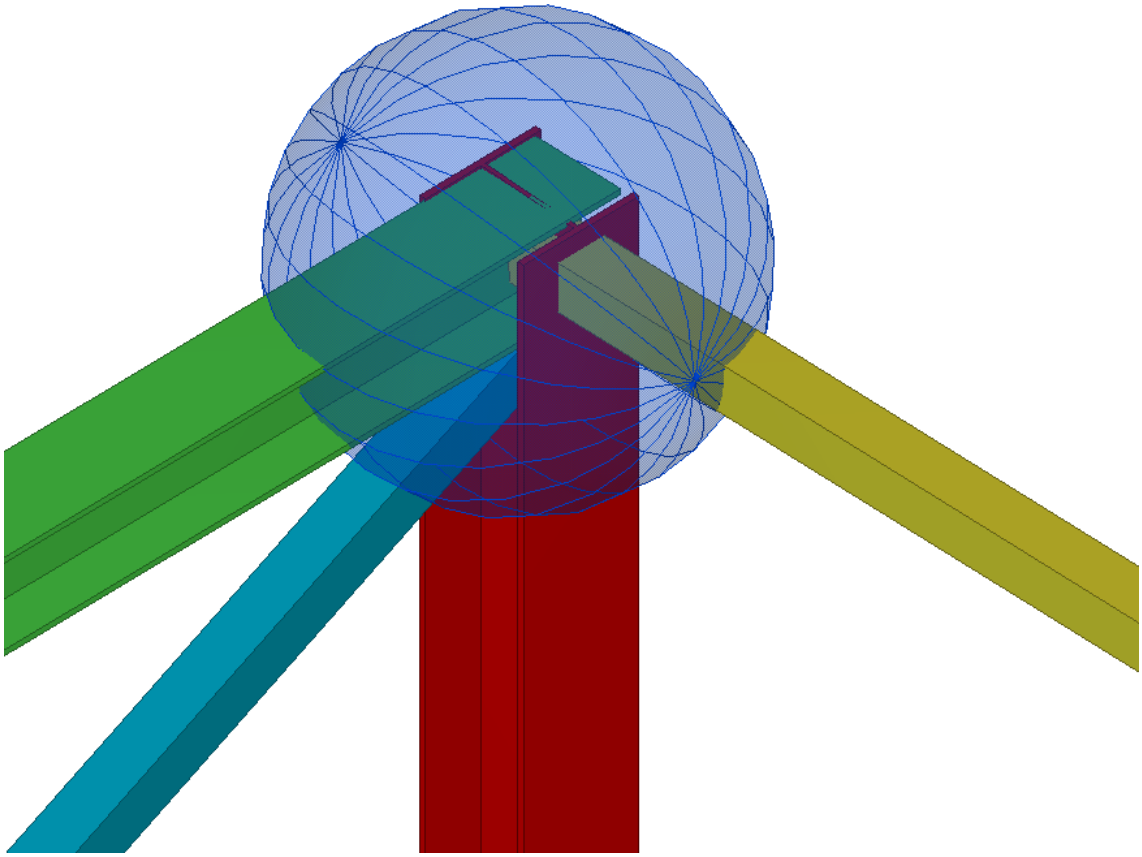

Figure 16. *An example of a detected connection area.*

In this study, labels were assigned by clustering using a decision tree where every unique connection got their own label. Clustering was done analyzing properties of parts and how they were attached to structural members. Assigning a label to every unique connection allows program to propose multiple options for each connection area rather than predicting only one suitable connection. Proposing multiple connections would have not been the case if labels were assigned by analyzing properties and relationships of connections that are used for finding connections.

3.2.2 Preprocessing

Most of the information processed by BIM software tools is structured and hierarchical. The data consist of predefined types of objects where each type has in most cases the same properties. However, the properties may have other objects with their own properties. For example, an assembly may contain multiple parts and each of these parts contains their own properties.

Generally, hierarchical data is a typical problem for machine learning algorithm. Machine learning datasets are most commonly tables with columns corresponding to a single at-

tribute and rows corresponding to a single observation (Bowles, 2015). Generating suitable, one dimensional feature vectors of connections, is a complicated task. For that reason, the data preparation consisted of three steps: selecting relevant data, preprocessing data and transforming data.

In comparison to creating database, where all data was included, only meaningful properties for finding connections were selected. After analyzing the data, the feature vectors were built from connection areas and they contained geometrical information about structural members and BIM properties. With these properties, it was possible to cover most of the rules used to select the connections to be implemented in the BIM.

Commonly machine learning algorithms are working with numerical data although algorithms for different data types exist. However, data, with mixed string and number values, needs preprocessing and commonly string values are transformed to numerical values. As it was explained earlier, Euclidian space was selected to be used with the algorithm and thus the need for conversion. Moreover, in Euclidian space, a distance between samples matters and for that reason, all features ranges were normalized between 0 and 1.

3.2.3 Algorithm

The algorithm had one goal: predict the best matching connections between structural members. For finding the best suitable connections, information how a connection is built and what parts are used in it are not relevant. Instead of including information about connections, the best suitable connections are found by their connection areas. While connections differ dramatically, the connection areas are relatively similar and contain enough data for modelling connections to them.

As told above, the NN approach suits well for finding best matches. For that reason, k-nearest neighbors (k-NN), where the Euclidian distance metric is used to measure distances between examples, was used in this study. k-NN is a classifying algorithm that classifies each example based on the majority of k-nearest neighbors in the training set (Weinberger et al., 2006). This method is widely used in different areas, such as pattern recognition and data mining (Shakhnarovich et al., 2006).

According to Hastie et al. (2009), NN methods use those observations in the training set closest in input space to x to form prediction of output $\hat{Y}$. k-NN can be defined as follows (Hastie et al., 2009):

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i, \quad (2)$$

where $N_k(x)$ is the neighborhood of x defined by the k closest points to x_i in training sample. Figure 17 shows an example of k-NN binary classification, where used k was 15. The colored regions indicate on which class points in input space will be classified.

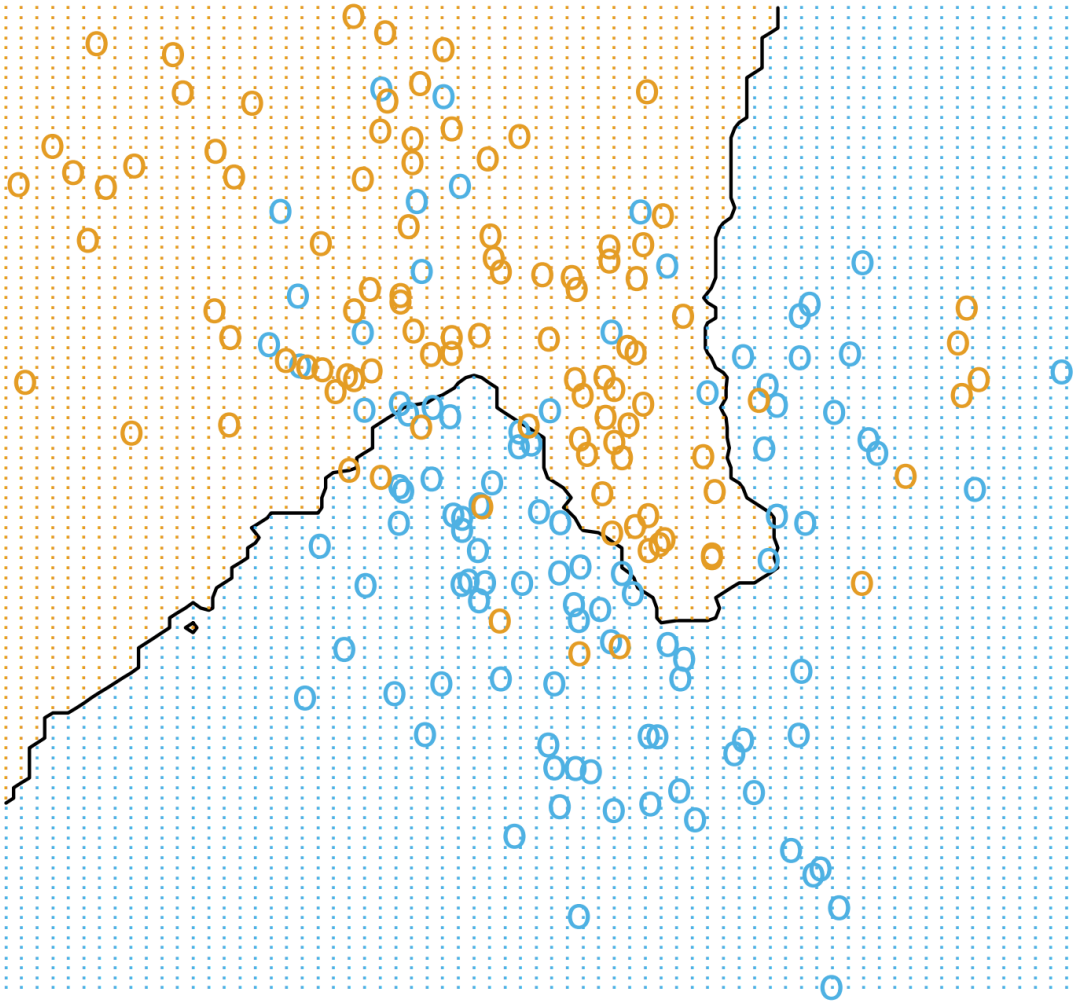

Figure 17. *The k-NN binary classification example in two dimensions, where k is 15. (Hastie et al, 2009).*

The main perk to use k-NN is its simplicity and efficiency (Zhang et al., 2017). While being one of the simplest machine learning algorithms, it can be very powerful in correct situations. Furthermore, k-NN has shown remarkable performances on data with a large example size (Zhang et al., 2017). However, selecting the k may affect to the performance (Zhang et al., 2017).

The algorithm was implemented with Python. Python is one of the most popular programming languages for scientific computing and it is an appealing choice for data analysis (Pedregosa et al., 2011). In this study, machine learning algorithms were imported from Scikit-learn. Scikit-learn is a Python module that includes a wide range of machine learning algorithms for both supervised and unsupervised problems (Pedregosa et al., 2011).

Moreover, Python modules can be run by C# modules which removed any obstacle to use Python.

3.2.4 Reporting results

The way to present results depends on people who deal with the results (Bell, 2014). Usually results are shown in simple format such as in a spread sheet or in a database table (Bell, 2014), but in this case, information in text format do not tell enough. To improve toolkits usability, the results are shown visually.

The presenting results contains two parts: visualizing connection areas and showing the best matching connections. Visualizing connection areas is done by creating an indicator for each area. Example of found connection area and way it is visualized can be seen in Figure 16.

Secondly, a snapshot from the connection area and parts is shown to the users. Example of the snapshot can be seen in Figure 18. While creating a database, a three-dimensional snapshot was taken from the connections including all the members and parts in the connection. After classification, three-dimensional snapshots of the best matches and information how well matches fits to a connection area were shown the users.

Assigning labels by clustering enabled the possibility to propose the most used connections for each connection area. The connections shown in UI are sorted first by estimated accuracy the proposed connection fits in that connection area and then by times the connection is used. Every time a connection is selected and inserted into a model, it, the toolkit will promote it more. This helps manufacturing parts to these connections, as the parts are more similar than in cases where each designer selects connections by their own preferences.

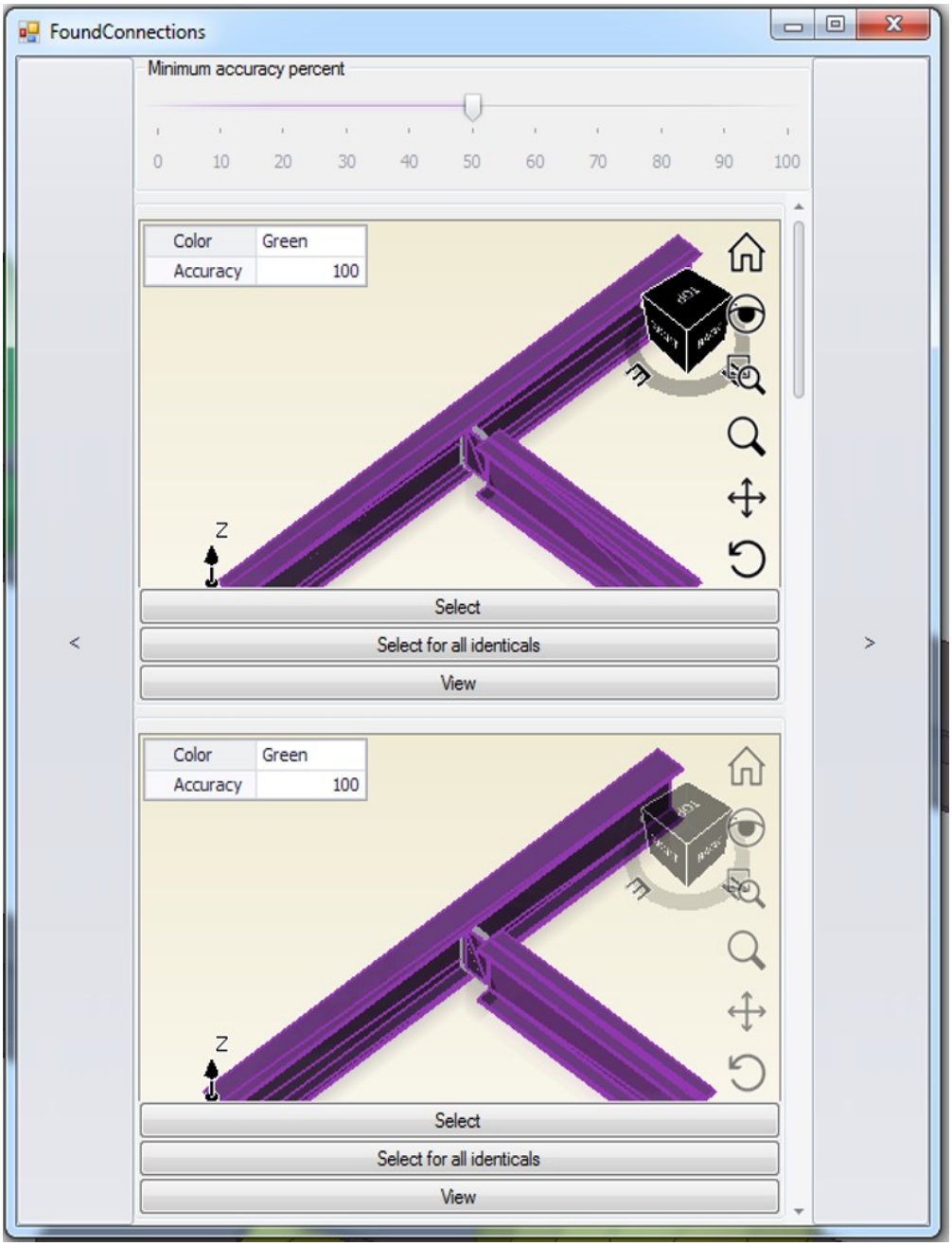

Figure 18. *An example of proposed connections to a connection area*

4. RESULTS

4.1 Methods and objectives

The main objective of this thesis is to make an implementation of toolkit to automate steel connections design in BIM softwares by applying machine learning techniques. For analyzing how well the toolkit can do the assigned task, the toolkit was tested two ways: how well the toolkit found connections to connections areas and is it possible to automatically model the found connections between structural members.

Finding connections for connection areas is crucial for this approach. If the connection areas do not have enough similarities between each other, it would not be possible to use this approach. Moreover, if it is not possible to find connections with used dataset, it raises the question if this approach is right and worth to continue investigating.

Modelling the found connections is critical for automation and for improving modelling efficiency. If it is not possible to fully model connections automatically or quality of automatically modelled connections is unacceptable, structural designers have to manually modify them. This would probably take more time and effort than creating the connections from the very beginning.

Finally, it is worth to investigate how it is possible to reuse the data in previously modelled BIM models. With the used approach, this data is used again, and it brings new possibilities for other approaches for reusing the data stored in BIM models. However, this study is scoped to show only the proof of concept of reusing the BIM data and other approaches to use the data are not investigated in this study.

4.2 Test cases

After creating the toolkit, the algorithms were tested with 13 industrial steel structures designed in Tekla Structures software, using 10 of them for creating datasets and searching connections to 3 of them. These buildings are located in various countries and have different industrial purposes. The models had different amount of connections that were previously designed by structural engineers.

Number of connections varied from 730 to 9407, with the average equaling 4277 connections. 10 BIM models were used for creating training sets and the models contained

a total of 45027 connections. Training sets were created in two ways: (i) creating a training set from a single model or, (ii) creating a training set containing a combination of training sets from multiple models.

The toolkit was tested to automatically find connections for 3 models without structural connections using different training sets. These BIM models, here named models I, II and III, are shown in Figure 19.

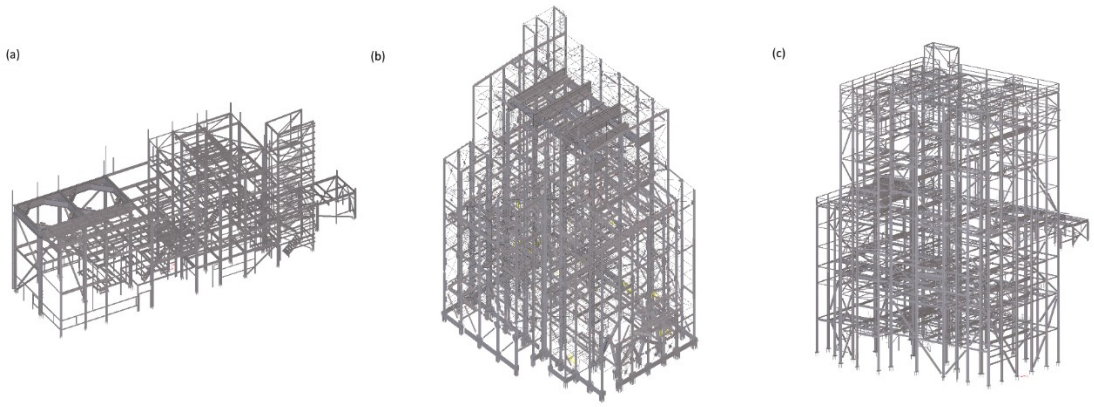

Figure 19. *Models for finding connections: (a) Model I; (b) Model II; (c) Model III*

Even though the toolkit was tested with 13 models, basically any steel structured building could be included. One of the benefits of using machine learning is possibility to add more connections to the database without sacrificing speed. In future, a goal is to add hundreds of models and their connections in the database.

4.3 Finding connections

To find out the similarity between connections of different BIM models, the connections found automatically were categorized using the best matching connection cluster. Equation (3) describes how the success percentage *p* was calculated from the connections distance to its closest matching cluster:

$$p = 100\ \% * (1 - d), \tag{3}$$

where *d* is the distance to closest matching cluster.

The success percentage was categorized as follows: ‘perfect match’, ‘possible match’ and ‘no match’. Although users can decide ranges for these categories, during testing the ranges were kept constants. Perfect matches contain only matches with 100% success percentage, possible matches are those where 100% > p ≥ 50% and no matches

has a p < 50%. These percentage limits were selected, after finding the worst success percentage that resulted in a valid structural connection.

The perfect match connections created automatically in the new BIM model are the same as those found from the training sets. However, connections created in new BIM models that are possible matches do not have identical parameters as those designed in previous BIM models. These differences may be due to the geometrical relationship between structural members, angles between members, or in BIM properties not affecting the structural integrity of the connection.

Example of a connection finding process can be seen in Figure 20. On the left image is the model after connection areas are searched and connection area indicators are inserted. On the right image, the same connection areas are colored after searching connections according to the success percentage category. Green areas represent perfect matches, yellow are possible matches, and red are connection areas where the toolkit failed to find matches.

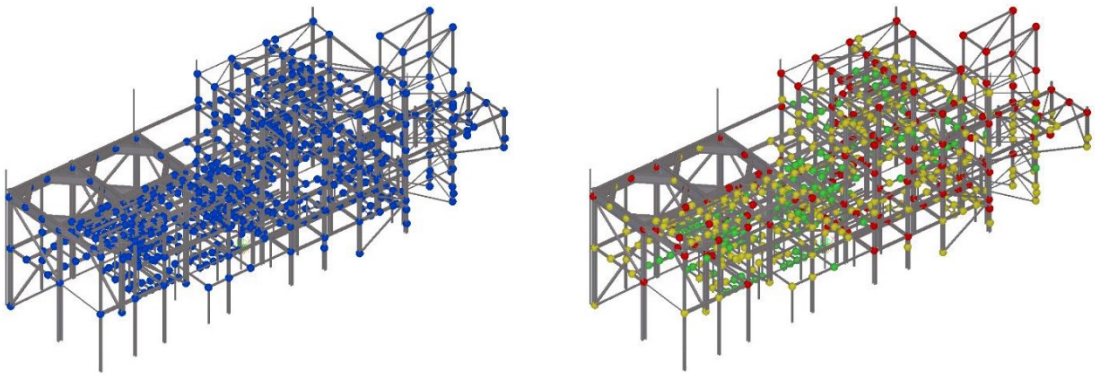

Figure 20. *Representation of the results of the toolkit in BIM model I: (Left) Connection areas; (Right) Results presented using colors.*

The algorithm searched connections by: (i) creating a training set from a single model or, (ii) creating cumulatively a combined training set which included previously used single model training sets. Results were measured in two ways: percentages of perfect matches and percentages of perfect and possible matches. The results are visible in Table 1 and Table 2.

Table 1. Percentages of perfect matches for model I, model II, and model III.

	Model I		Model II		Model III	
	Single	Combined	Single	Combined	Single	Combined
Model 1	0.00 %	0.00 %	0.00 %	0.00 %	0.02 %	0.02 %
Model 2	0.69 %	0.69 %	0.00 %	0.00 %	0.00 %	0.02 %
Model 3	13.18 %	13.18 %	0.54 %	0.54 %	0.54 %	0.56 %
Model 4	15.81 %	18.72 %	0.13 %	0.54 %	0.17 %	0.71 %
Model 5	6.38 %	20.53 %	0.05 %	0.66 %	0.05 %	0.75 %
Model 6	3.19 %	21.50 %	0.00 %	0.66 %	6.57 %	7.08 %
Model 7	18.72 %	24.69 %	0.29 %	0.68 %	1.41 %	8.24 %
Model 8	10.26 %	24.69 %	0.00 %	0.68 %	0.22 %	8.24 %
Model 9	14.70 %	26.21 %	0.32 %	0.72 %	0.90 %	8.73 %
Model 10	0.69 %	26.21 %	0.00 %	0.72 %	0.00 %	8.73 %

Table 2. Percentages of perfect and possible matches for model I, model II, and model III.

	Model I		Model II		Model III	
	Single	Combined	Single	Combined	Single	Combined
Model 1	54,37 %	54,37 %	44,51 %	44,51 %	43,82 %	43,85 %
Model 2	70,46 %	71,84 %	23,54 %	58,91 %	61,84 %	64,49 %
Model 3	56,87 %	73,09 %	25,06 %	62,57 %	62,57 %	66,73 %
Model 4	49,79 %	74,76 %	17,16 %	62,82 %	26,05 %	67,24 %
Model 5	44,80 %	76,01 %	17,41 %	63,11 %	24,54 %	68,02 %
Model 6	69,49 %	77,25 %	21,24 %	64,23 %	31,42 %	69,48 %
Model 7	53,95 %	78,36 %	22,23 %	65,26 %	63,74 %	72,32 %
Model 8	62,83 %	78,92 %	22,75 %	66,22 %	64,01 %	73,35 %
Model 9	57,56 %	79,47 %	21,71 %	67,03 %	36,48 %	74,61 %
Model 10	67,96 %	79,61 %	26,95 %	67,35 %	58,17 %	74,81 %

For visualizing the trends, the values in Table 1 and Table 2 are plotted in Figure 21, Figure 22, and Figure 23. These figures contain graphs for percentages of perfect matches and percentages of perfect and possible matches from training sets from single models and from cumulative combination of training sets from multiple models.

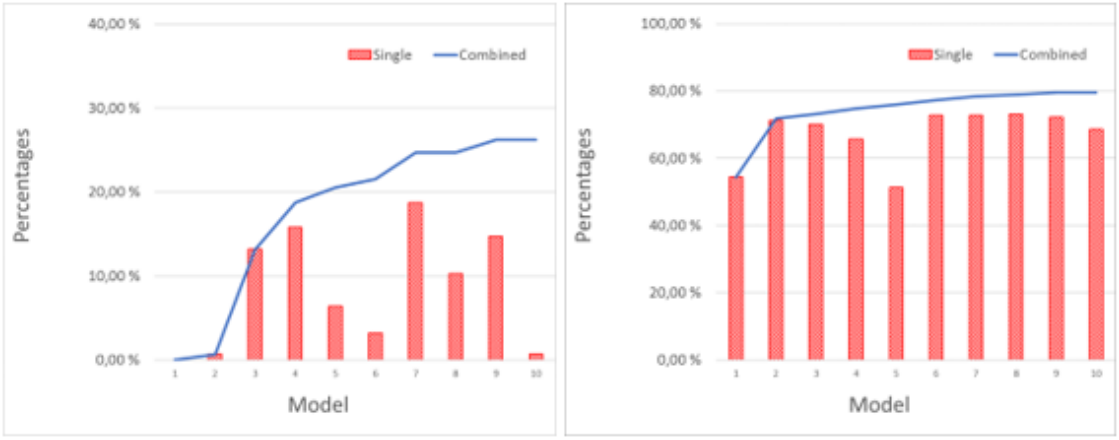

Figure 21. *Histogram and cumulative curve of the percentage of matches for model I: (Left) Perfect matches; (Right) Perfect and possible matches.*

For model I, the percentages of perfect matches from a single model variated between models from 0.0 % to 18.7 % (Figure 21; left). However, the percentage of perfect matches from combined datasets did raise to 26.1 %. The percentages of perfect and possible matches variated less than the percentages of perfect matches and all the percentages from single models were between 51.2 % and 72.7 % (Figure 21; right). Moreover, the percentage from datasets containing connections from multiple models kept raising after every model to 79.6 %.

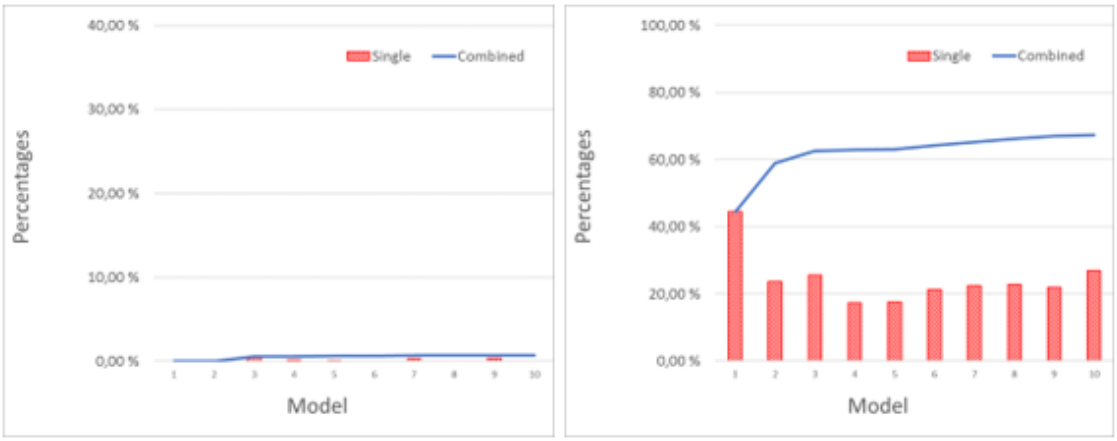

Figure 22. *Histogram and cumulative curve of the percentage of matches for model II: (Left) Perfect matches; (Right) Perfect and possible matches.*

The percentages of found connections were worse for model II than for model I. The toolkit found close to no perfect matches. The number of found connections were between 0 and 30 connections and the percentages from single model were between 0.0 % and 0.54 % (Figure 22; left). Moreover, the percentage from combined datasets stayed low and it was at the end 0.7 %. However, the percentage of perfect and possible matches from connection from multiple models reached 67.4 % where the percentages from single models were between 17.3 % and 44.5 % (Figure 22; right).

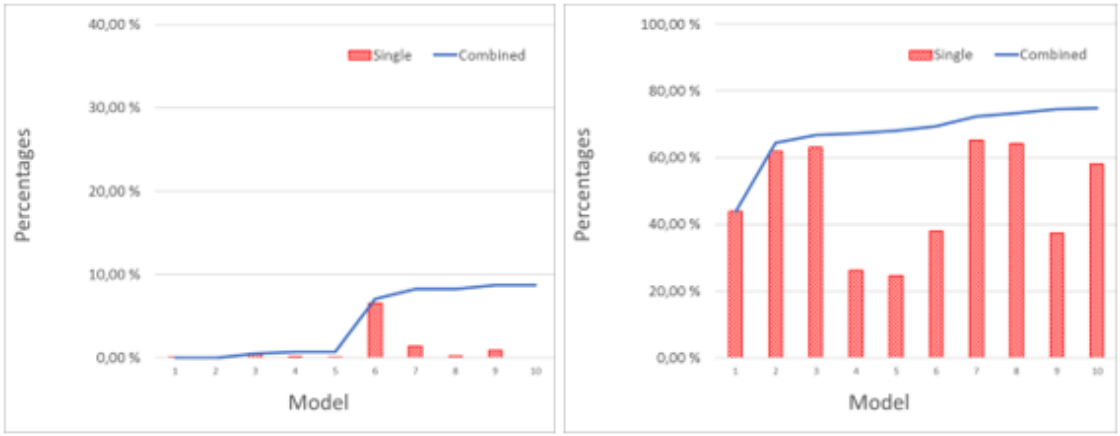

Figure 23. *Histogram and cumulative curve of the percentage of matches for model III: (Left) Perfect matches; (Right) Perfect and possible matches.*

For model III, the percentages of perfect matches from single models was also worse than for model I, where the percentages were less than 2 % from all models except from one and the percentage from a dataset containing connections from all models was 8.7 % at the end (Figure 23; left). The percentages of perfect and possible matches from single models were between 24.6 % and 65.2 % (Figure 23; right).

4.4 Automatic modelling

For testing, the best matching connection found for each connection area was automatically inserted to the BIM model. The connections were modelled by plugins or by native Tekla Structures parts depending on whether original connections were modelled by plugins or by primitive objects. As previously discussed, it is considerably faster to make changes to plugins than changing connections created by combining primitive parts.

With the perfect matches, created connections were identical. They had same properties and geometry than the original connection and it was not possible to find difference between automatically inserted connections and connections in original models. When the success percentage was less than 100 % and geometrical properties or relationships between members were not identical to original connections, needed geometrical properties of modelled connections were automatically modified and the connections differed from the original connections.

4.5 Quality

The quality of automatically created connections can be validated in two steps: quality of proposed connections and quality of automatic modelling.

Typically, classification algorithms are validated by having a separate test dataset. As shown in Figure 9, this dataset contains the same information than the training dataset and correct labels. By comparing predicted labels to correct labels, an accuracy to predict correct label can calculated and trained models can be validated.

As mentioned, typical practices to validate trained models require labelled test data. Furthermore, each instance of a test dataset is required to have only a single label, or each instance of a test dataset is required to contain all the possible labels. There are numerous ways to design structurally working connections between structural members and, as mentioned earlier, every unique connection got their own label.

Therefore, it is not reasonable to compare predicted and used connections since typically there is not just a single suitable connection. Moreover, in scope of this thesis, it was not possible to manually set all possible connections to every connection area to which connections were used to find. For that reason, the quality of prediction is not analyzed and the accuracy to predict used connections are not included in this thesis.

The quality of automatically created connections was assessed by analyzing the collisions with other neighbor objects and how they were linked to the structural members. Because of the number of automatically modelled connections, it was decided that the quality of modelled connections needs to be evaluated automatically. Normally, BIM softwares provide a way to detect clashes between objects and how parts are connected to each other. This information was used for analyzing the quality.

5. DISCUSSION

The results were positive, even being limited by using only a 10-model database. Finding similarities in connections is critical for automatically creating steel connections in BIM models. When the models are sufficiently similar, it is possible to find patterns between the connections and between the corresponding members being connected. This study has shown that steel connections did share similarities between models used in testing. Due to this, it was possible to automatically insert connections in the models.

5.1 Evaluation

Interestingly, as evidenced by Table 1 and Table 2, it was found that the number of matching connections varies between the models. Although all the test models were industrial buildings, some models had no common connection between them. However, usages of the buildings differ, and their structures are optimized for their usages, and therefore they share only some similar solutions.

The number of connections with possible matches was considerably larger than the number of connections with perfect matches. Percentages of perfect and possible matches for combined training dataset containing connections from all 10 test models did not vary greatly and they were between 67,35 % and 79,61 %.

However, the models had relatively few connections with perfect matches. One BIM model can have less than 1 % of the connections with perfect matches (Figure 22; left), but still find possible connection to over 65 % of all the connections (Figure 22; right). Moreover, percentages of perfect matches variated more than the percentages of perfect and possible matches. As evidenced in Table 1, two models can have 18,72 % of all connections identical but in worst cases, there is not a single identical connection between models.

The increase in both perfect matches and combined perfect and possible matches suggests that increasing the number of models and, thereby, the number of connections in the training sets, will allow the algorithms to find more and better matchings. This was expected when applying machine learning techniques. In any case, the results were positive, even being limited by using only a 10-model database. Importantly, the toolkit was able to automatically model connections in models with no previous detailed connections.

We expect that it is possible to reduce detailing time in BIM projects much further, by enhancing the efficiency of the present methods. Since that was not included to the scope of this thesis, investigation of efficiency is an important future next step. By comparing detailing time with present methods to needed time with proposed approach, it would possible to gain more knowledge if it is possible to improve efficiency of designing process with this approach.

5.2 Limitation and reliability

Using these automation methods does not directly mean that detailing BIM models would necessarily be more efficient. If the quality of automatically modelled connections is not sufficient, the engineers should manually enhance them. It is the speed of this enhancement that will determine whether this new methodology can be less time consuming than traditional methods.

Also, as addressed previously, the quality of proposed connections and the quality of modelled connections were not verified in greater detail and needs further investigation. An important future next step is to use this toolkit and use these methods in real BIM projects, during which the methods will be further tuned. E.g., the quality of predicted connections could be improved by further tuning the input parameters for the machine learning methods or by refining processes used by these algorithms.

Another issue for future development is better analyzes to capacities. The current prototype is not calculating the capacities, nor they can be manually set. Suggesting connection with similar member loads leaves possibility that the toolkit will suggest connections that will break under loads they are planned to transfer. However, designers have to do capacity checks whether they are using this toolkit or not and this may not be adding more tasks to designers.

Finally, there is no way to tell if a modelled connection is modelled correctly. It can be estimated by collisions and how parts are attached to each other. Otherwise, there is no automated way to tell precisely if a modelled connection contains inaccuracies or mistakes. Furthermore, it is possible that connections read to the database contain faults or they are modelled wrong. Both of these issues are handled by having multilevel verification which is already used in designing.

5.3 Theoretical contribution

While doing a background check, it became clear that machine learning is not widely researched in the AEC industry. That is surprising because the construction industry is

renowned for its poor productivity and lags behind other industries in the rate by which improvements are introduced (Fulford and Standing 2014; Segerstedt et al., 2010). It has been said that BIM modelling might not be able to take advantage of machine learning techniques (Correa, 2015). Our results seem to disprove this. Nonetheless, Correa agrees that the AEC industry could get enormous benefits from analyzing BIM data.

Currently, BIM softwares are missing utilization of the data stored in previously modelled BIM models. Typically, when a construction project is finished, the BIM model is not actively utilized anymore. Even though structural engineers spend hours to think different solutions and even more hours to model those ideas, these are not being reused. Keeping that in mind, it is surprising that machine learning is not more used to learn from those models. Typically, BIM contains a huge number of repetitive tasks, which could be automatized using machine learning.

The automatic generation of BIM models is not a new idea and it has generally been explored (Banfi et al., 2017; Eastman et al., 2009; Wang et al., 2015), but usually they are not using machine learning for that. Although machine learning is widely used in many fields, ranging from Computer Science, to Physics and Biology, researches and use cases of machine learning are lacking in the AEC industry. Nevertheless, it has said that it will become part of the AEC industry at some point (Bloch & Sacks, 2018).

5.4 Further research

As mentioned previously, there are multiple ways to continue research on this subject and they can roughly be divided into three group: test in real projects, continue developing the toolkit, and expand the coverage of the toolkit.

The most natural way to continue research is to test the toolkit in real projects. By comparing designing and its efficiency between a typical design process and a design process where the toolkit is used, it is possible to tell if the toolkit actually increases modelling speed. Improving designing efficiency as well as improving quality of designs are the main reasons to implement new ways to do designing and it is crucial that these ways actual are improving designing without sacrificing efficiency or quality. Additionally, testing in real projects is needed for validating predicted connections.

The development of the toolkit can be continued, and new features can be added. For example, including strength calculation will improve accuracy of capacities and changing to more sophisticated classification algorithm may improve accuracy of predictions. However, analyzing accuracy of different algorithms can’t be done unless the toolkit is tested in real projects.

Furthermore, a possibility to propose connections allows development for better prioritized designs. Each connection in a database could have different criterions, such as material consumption, cost, speed of installation or CO2 emission, and connections are chosen based on values of clients.

Finally, it is worth to investigate if this approach works with other BIM softwares than Tekla Structures. It can be assumed that there are no obstacles to use same approach with other BIM softwares as the toolkit was built to be BIM software independent. Also, the toolkit could be expanded to work with other material, such as concrete or timber. However, most of these potential future researches are natural directions to develop the toolkit and after promising results, they are likely implemented to the toolkit.

6. CONCLUSIONS

As shown by previous studies, automation by machine learning can be implemented in BIM processes. The purpose of this study was to develop a toolkit for more efficient design of steel connections in BIM models. The proposed approach was based on classifying structural steel connections with k-NN to automate the design. Results indicate that our strategy provides a new solution for the scope of BIM in structural engineering design.

This study has shown that steel connections did share similarities between models used in testing. Finding similarities in connections is critical for automated creation of steel connections in BIM models. When the models are sufficiently similar, it is possible to find patterns between the connections and between the corresponding members being connected. Due to this, it was possible automatically insert connections in the models.

Importantly, information stored in previous BIM models were gathered and it was used to automatically insert connections in BIM models without structural connections. The observed and the potential benefits in modelling structural connections with this and similar methods makes this subject worth investigating further, since the results are expected to enhance the traditionally low productivity of the construction industry.

REFERENCES

Banfi, F., Fai, S. and Brumana, R. (2017). BIM Automation: advanced modeling generative process for complex structures, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences.

Bell, J. (2014). Machine Learning: Hands-On for Developers and Technical Professionals. John Wiley & Sons.

Bloch, T. and Sacks, R. (2018). Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models, Automation in Construction, 91 (6), 256-272.

Bowles, M. (2015). Machine Learning in Python: Essential Techniques for Predictive Analysis. John Wiley & Sons.

Butterfield, A. and Ngondi, G.E. (2016). A Dictionary of Computer Science (7th ed.). Oxford University Press.

Chervonenkis A.Y. (2011). Problems of Machine Learning, Lecture Notes in Computer Science, 6744, 21-23.

Chi, HL., Wang, X. and Jiao, Y. (2015). BIM-Enabled Structural Design: Impacts and Future Developments in Structural Modelling, Analysis and Optimisation Processes. Archives of Computational Methods in Engineering, 22 (1), 135-151.

Corenc, B.E., Tinyou R. and Syam, A. (2005). Steel designers' handbook (8th ed.). NewSouth Publishing.

Correa, F.R. (2015). Is BIM Big Enough to Take Advantage of Big Data Analytics? ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, 32, 1-8.

Dastbaz, M., Gorse, C. and Moncaster A. (2017). Building Information Modelling, Building Performance, Design and Smart Construction. Springer.

Davis E. and Marcus G. (2015). Commonsense reasoning and commonsense knowledge in artificial intelligence. Commununications of the ACM, 58 (9), 92–103.

Davison, B. and Owens, G.W. (2011). Steel Designers' Manual. John Wiley & Sons.

Eastman, C., Lee, J., Jeong, Y. and Lee, J. (2009). Automatic rule-based checking of building designs, Automation in Construction, 18 (8), 1011-1033.

Eastman, C., Teicholz, P., Sacks, R. and Liston, K. (2011). BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers, and Contractors (2nd ed.). John Wiley & Sons.

Finne, C., Hakkarainen, M., and Malleson, A. (2013). Finnish BIM Survey 2013

Flach, P. (2012). Machine Learning: The Art and Science of Algorithms that Make Sense of Data. Cambridge University Press.

Fulford, R. and Standing, C. (2014). Construction industry productivity and the potential for collaborative practice, Internal Journal of Project Manage, 32 (2), 315–326.

Han, J., Kramber, M. and Pei, J. (2006). Data Mining: Concepts and Techniques (2nd ed.). Elsevier.

Han, J., Kramber, M. and Pei, J. (2011). Data Mining: Concepts and Techniques (3rd ed.). Elsevier.

Harding, J., and Shephard, P. (2017). Meta-Parametric Design, Design Studies, 52 (9), 73-95.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). The elements of statistical learning: data mining, inference, and prediction (2nd ed.). Springer.

Heinisuo, M., Laasonen, M., Ronni, H., and Anttila, T. (2010). Integration of joint design of steel structures using product model. Proceedings of the Computing in Civil and Building Engineering, 323.

Helminen, J., Tuori, J., Laasonen, M., and Farinha, R. (2018). Automated Generation of Steel Connections of BIM by Machine Learning. 17th International Conference on Computing in Civil and Building Engineering, 631-636.

Jain, A.K. (2010). Data clustering: 50 years beyond K-means, Pattern Recognition Letters, 31 (8), 651-666.

Kensek, K., and Noble, D. (2014). Building Information Modelling: BIM in Current and Future Practice. John Wiley & Sons.

Ky, J. (2013). C#: A Beginner’s Tutorial. Brainy Software.

Lalla, A. (2017). Kantavien rakenteiden parametrinen suunnittelu ja mallintaminen. (Master of Science Thesis)

Liu, X.C., Pu, S.H., Zhang, A.L., Xu, A.X., Ni, Z., Sun, Y. and Ma, L. (2015). Static and seismic experiment for bolted-welded joint in modularized prefabricated steel structure. Journal of Constructional Steel Research, 115 (12), 417-433.

Murphy, K. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.

NIBS. (2007). National building information, modeling standards part-1: overview, principles and methodologies. US National Institute of Building Sciences Facilities Information Council, BIM Committee.

Ning, G. and London, K. (2010). Understanding and facilitating BIM adoption in the AEC industry, Automation in Construction, 19 (8), 988-999.

Oti, A.H., Tizani, W., Abanda, F.H., Jaly-Zada, A. and Tah, J.H.M. (2016). Structural sustainability appraisal in BIM, Automation in Construction, 69 (9), 44-58.

Owens, G.W. and Cheal, B.D. (1989). Structural Steelwork Connections. Butterworth & Co.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12, 2825-2830.

Rugarli, P. (2018). Steel Connection Analysis. John Wiley & Sons.

Sacks, R., Eastman C.M. and Lee, C. (2004). Parametric 3D modeling in building construction with examples from precast concrete. Automation in Construction, 13 (3), 291-312.

Shakhnarovich, G., Darrell, T. and Indyk, P. (2006). Nearest-Neighbor Methods in Learning and Vision: Theory and Practice. MIT Press.

Segerstedt, A., Olofsson, T., Bankvall, L., Bygballe, L. E., Dubois, A. and Jahre, M. (2010). Interdependence in supply chains and projects in construction, Supply Chain Management, 15 (5), 347–353.

Tekla Open API. Retrieved from Tekla Open API website: https://www.tekla.com/open-API, accessed on February 9, 2018.

Tekla Structures Glossary. Retrieved form Tekla Structures Support website: https://teklastructures.support.tekla.com/system/files/files/TeklaStructuresGlossary.pdf, accessed on May 15, 2018.

Volk, R., Stengel J. and Schultmann, F. (2014). Building Information Modeling (BIM) for existing buildings — Literature review and future needs, Automation in Construction, 38 (3), 109-127.

Voznika, F. and Viana, L. (2007). Data Mining Classification. Retrieved from https://courses.cs.washington.edu/courses/csep521/07wi/prj/leonardo_fabricio.pdf, accessed on June 15, 2018.

Wang, C., Cho, Y.K. and Kim, C. (2015). Automatic BIM component extraction from point clouds of existing buildings for sustainability applications, Automation in Construction, 56 (8), 1-13.

Weygant, R. (2011). BIM Content Development: Standards, Strategies, and Best Practices. John Wiley & Sons.

Witten, I. H., Frank, E., and Hall, M. A. (2011). Data Mining : Pratical Machine Learning Tools and Techniques (3rd ed.). Elsevier.

Yarmohammadi, S., Pourabolghasem, R. and Castro-Lacouture, D. (2017). Mining implicit 3D modeling patterns from unstructured temporal BIM log text data, Automation in Construction, 81 (9), 17-24.

Zhang, S., Li, X., Zong, M., Zhu, X., and Cheng, D. (2017). Learning k for kNN Classification, ACM Transactions on Intelligent Systems and Technology, 8 (3), Article 43.

Zheng, W., Zhao, L., and Zou, C. (2004). Locally nearest neighbor classifiers for pattern classification, Pattern Recognition, 37 (6), 1307-1309.